Intelligent-Agent-Based Management of Heterogeneous Networks for the Army Enterprise

Richards, Clyde E., Jr.

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

INTELLIGENT-AGENT-BASED MANAGEMENT OF HETEROGENEOUS NETWORKS FOR THE ARMY ENTERPRISE

by

Clyde E. Richards Jr.

September 2003

Thesis Advisor:	Alex Bortdetsky
Second Reader:	James O'Donnell

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2003	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE: Intelligent-Agent-Based Management of Heterogeneous Networks for the Army Enterprise			**5. FUNDING NUMBERS**
6. AUTHOR(S) Clyde E. Richards			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited			**12b. DISTRIBUTION CODE**

13. ABSTRACT ***(maximum 200 words)***

The Army is undergoing a major realignment in accordance with the Joint Vision 2010/2020 transformation to establish an enterprise command that is the single authority to operate and manage the Army Enterprise Information Infrastructure (Infostructure). However, there are a number of critical network management issues that the Army will have to overcome before attaining the full capabilities to manage the full spectrum of Army networks at the enterprise level. The Army network environment consists of an excessive number of heterogeneous applications, systems, and network architectures that are incompatible. There are a number of legacy systems and proprietary platforms. Most of the NM architectures in the Army are based on traditional centralized NM approaches such as the Simple Network Management Protocol (SNMP). Although SNMP is the most pervasive protocol, it lacks the scalability, reliability, flexibility and adaptability necessary to effectively support an enterprise network as large and complex as the Army. Attempting to scale these technologies to this magnitude can be extremely difficult and very costly. This thesis makes the argument that intelligent-agent-based technologies are a leading solution, among the other current technologies, to achieve the Army's enterprise network management goals.

14. SUBJECT TERMS Intelligent Agent, SNMP, Enterprise Network Management, CoABS, Army Enterprise Infostructure, Global Information Grid			**15. NUMBER OF PAGES** 141
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release, distribution is unlimited.

INTELLIGENT-AGENT-BASED MANAGEMENT OF HETEROGENEOUS NETWORKS FOR THE ARMY ENTERPRISE

Clyde E. Richards Jr.
Major, United States Army
B.A., Rutgers University, 1989

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
September 2003

Author: Clyde E. Richards Jr.

Approved by: Dr. Alex Bordetsky
Thesis Advisor

Mr. James O'Donnell
Second Reader/Co-Advisor

Dr. Dan Boger
Chairman, Information Systems Academic Group

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The Army is undergoing a major realignment in accordance with the Joint Vision 2010/2020 transformation to establish an enterprise command that is the single authority to operate and manage the Army Enterprise Information Infrastructure (Infostructure). However, there are a number of critical network management issues that the Army will have to overcome before attaining the full capabilities to manage the full spectrum of Army networks at the enterprise level. The Army network environment consists of an excessive number of heterogeneous applications, systems, and network architectures that are incompatible. There are a number of legacy systems and proprietary platforms. Most of the NM architectures in the Army are based on traditional centralized NM approaches such as the Simple Network Management Protocol (SNMP). Although SNMP is the most pervasive protocol, it lacks the scalability, reliability, flexibility and adaptability necessary to effectively support an enterprise network as large and complex as the Army. Attempting to scale these technologies to this magnitude can be extremely difficult and very costly. This thesis makes the argument that intelligent-agent-based technologies are a leading solution, among the other current technologies, to achieve the Army's enterprise network management goals.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I firmly believe that we exist on this earth to obtain the knowledge and wisdom of God the creator in order to achieve salvation. However, salvation is only granted to those that pursue it. An academic endeavor such as this is an exercise in discipline and learning. It better conditions our minds to discern, synthesize, and assimilate in order to better understand the wisdom of God. Some are more naturally inclined than others, but anyone with "faith" and the "will" to learn can obtain God's wisdom regardless of their natural abilities. I say this because I must put this thesis and my learning experience in the proper perspective. Thus, I humbly acknowledge God and praise Him for granting me the faith and will to obtain this academic achievement that has culminated in this thesis. Without Him, it just simply would not have been possible. I will certainly apply these hard-earned, privileged abilities to my salvation journey and to support the journey of others. THANK YOU GOD!

I must also thank those who have supported and helped me to succeed. I thank my dear wife Eleonore and my beautiful kids, Camille and Clyde III, for putting up with my uneasy temperament during the stressful times, and for picking up my weight when I needed it most to keep me sane – I LOVE YOU! I must also thank my Mom and other family members, my friends and the faculty for believing in me and supporting me. Special thanks go to Sam Chance, Marty Hagenston, and Doug Horner for getting me into this mess. You all played an important role in helping me prepare for this – THANK YOU! Father Michael Drury and all those that prayed for me, your prayers were not in vain – THANK YOU! Finally, I am grateful for Dr. Alex Bordetsky, my thesis advisor, and Jim O'Donnell, my second reader, for kindly accepting to assist my efforts – THANK YOU!

I humbly wish many blessings to all.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

…the network will become a weapon system and should have a command relationship commensurate with that of normal operational forces…

- BG (P) James D. Bryan, USA Commander, JTF-CND

A. DISCUSSION

The thesis of this paper is the argument that intelligent-agent-based technologies are a leading solution, among other current technologies, to achieve the Army's enterprise network management goals. The Army is undergoing a major realignment in accordance with the Joint Vision 2010/2020 transformation to establish an enterprise command that is the single authority to operate and manage the Army Enterprise Information Infrastructure (Infostructure). However, there are a number of critical network management issues that the Army will have to overcome before attaining the full capabilities to manage the full spectrum of Army networks at the enterprise level. Over the years the Army information infrastructure had evolved into a number of stovepiped networks, contemporary and legacy systems, and heterogeneous applications due to the lack of centralized configuration management and control.

The Department of Defense's effort to enable the overarching JV 2010/2020 concepts is the driving force behind the Army's need to establish a single enterprise-level network management architecture. JV 2010/2020 envisions the development of a superior joint force that is capable of achieving full spectrum dominance across the range of military operations. The pathway to full spectrum dominance is the underlying layered concepts of decision superiority, information superiority, network-centric warfare (NCW), and the global information grid (GIG). Each of the respective concepts, starting with the GIG, provides a distinctive capability and is the foundation that supports the preceding layer. These individual layers are the building blocks that enable full spectrum dominance.

Information superiority is the key enabler to achieve decision superiority for the warfighter, which ultimately leads to full spectrum dominance. Further, information superiority is enabled by the NCW concept, which requires the aggregation and

interoperability of the stovepiped networks, legacy systems, and applications. The GIG is the underlying infrastructure that supports NCW. The Army's portion of the GIG is called the Army Enterprise Infostructure (AEI). As can be seen, network management (NM) plays a vital role to realizing the JV2010/2020 concept. Through effective NM, networks must provide the necessary bandwidth availability, reliability, and quality of service for information exchange in an accurate, timely, and secure fashion.

There are a number of obstacles that the Army will have to overcome before achieving an effective enterprise NM solution. The Army network environment consists of an excessive number of heterogeneous applications, systems, and network architectures that are incompatible. There are a number of legacy systems that hinder interoperability. There are a number of proprietary platforms, including NM platforms. The NM platforms are based on different protocols and standards. Most of the NM architectures are based on traditional centralized NM approaches such as the Simple Network Management Protocol (SNMP), and the Common Management Information Protocol (CMIP). Although SNMP and CMIP are the most pervasive protocols, these standards apply agent technology in a very primitive way. Although advancements were made, such as in SNMP version 3 (SNMPv3), these protocols still lack the scalability, reliability, and adaptability necessary to effectively support an enterprise network as large and complex as the Army. Attempting to scale these technologies to this magnitude can be extremely difficult and very costly. This leads to the main research question: what alternative technologies can scale and meet the Army Enterprise Infostructure network management and situational awareness requirements?

This study proposes that intelligent agent technologies are a future leading solution to address the aforementioned problems. Although agent technologies solutions for network management are fairly immature, there are a number of studies that indicate they are promising. Agent-based technology has the capability to distribute intelligence throughout the network and dynamically perform NM functions on an as needed basis. This provides efficiency and flexibility, and dramatically cuts down on bandwidth constriction and overloading on a single, central processor. Agents can be added to any agent environment "on the fly," and because of their small size can scale well.

Considering these properties, as well as others, makes intelligent-agent-based technologies an ideal solution for the AEI network management requirement.

B. RESEARCH QUESTIONS

1. What alternative technologies can scale and meet the Army Enterprise Infostructure (AEI) network management and situational awareness requirements?

Sub-research questions:

a. How can intelligent-agent-based technologies be used to establish network management control and network situational awareness of the AEI?

b. How can intelligent-agent-based technologies be used to execute Fault, Configuration, Accounting, Performance, and Security (FCAPS) management or establish a network common operational picture (NETCOP)?

c. How does intelligent-agent-based technology for enterprise network management compare to SNMP-based and other distributed management technologies?

d. Can an intelligent-agent-based network management architecture scale to support the AEI?

e. How can the Control of Agent Based Systems (CoABS) be leveraged to support an intelligent-agent-based enterprise-level network management architecture for the AEI?

C. SCOPE

The scope of this thesis covers why intelligent-agent-based systems are very well suited to meet the Army's AEI network management requirements and how an intelligent-agent-based system can be applied to the AEI to achieve enterprise network management and network situational awareness. This thesis looks at the capabilities required for enterprise network management and the shortcomings that the Army will have to overcome based on the disposition of the current systems and networks. The study addresses some of the problems that traditional and distributed network management protocols pose, and makes the argument for intelligent-agent-based network

management relative to these problem areas. Finally, the thesis presents a conceptual, high-level intelligent-agent-based network management design based on current research projects such as the Control of Agent Based Systems (CoABS) endeavor sponsored by DARPA.

D. METHODOLOGY

The nature of this thesis research is to explore the numerous intelligent-agent-based architectures that are currently under study and use this research to make the argument for an intelligent-agent-based solution as opposed to traditional and other methods. Due to the fact that this field of research is relatively immature, there are no full scale intelligent-agent-based architectures that have yet to be applied to an enterprise network management situation. Hence, this study draws on the theory, experimentation and findings of the various architectures that have been published. The study looks at the lineage (i.e. the new warfighting concepts) that has lead up to the enterprise network management issues and breakdowns the overarching goals to the specific needs for the AEI. This thesis investigates the Army's approach to establish enterprise network management and a network common operational picture. The thesis culminates with the presentation of a conceptual, high-level design that is based on techniques and designs that are currently being developed.

E. BENEFITS OF THE STUDY

This thesis research is directly applicable to the military's effort to manage the envisioned enterprise-level network grids. The research will generate thought provoking ideas and present areas of concern that must be addressed in order to fully achieve the JV2010/2020 vision. The Army and the other services can benefit from this research by considering the complications, brought out in the thesis, which will eventually be encountered with the implementation of current technologies. By considering the benefits of intelligent-agent-based technologies elaborated in the study, the Army can seek the technology as a future alternative that can mitigate the shortcomings of current technologies and provide a cost effective solution for the AEI.

This research is sponsored by the Army Information Systems Engineering Command (ISEC), located at Fort Huachuca, Arizona. ISEC is currently pursuing

solutions to support enterprise network management and network situational awareness. This thesis research will aid ISEC in their pursuit for the best solution.

THIS PAGE INTENTIONALLY LEFT BLANK

II. THE ENTERPRISE NETWORK MANAGEMENT DILEMMA

A. NETWORK MANAGEMENT OVERVIEW

The Simple Network Management Protocol (SNMP) is the most pervasive and commonly accepted and implemented network management (NM) standard today. The Department of Defense's (DoD's) Joint Technical Architecture (JTA) document mandates SNMP as the data communications management standard within the DoD. With the increasing size, management complexities, and service requirement of today's networks, the limitations of classic agent-manager paradigms, such as SNMP, are inadequate to achieve the order of magnitude demands required in large organizations such as the DoD.

In the pursuit of achieving a single enterprise-level network across the military services, the DoD and the services will discover inherent limitations of the modern day network management protocols. This section reviews the network management concepts and the problems and issues that it poses for enterprise network management implementation envisioned within DoD.

1. Enterprise Networks

Enterprise networks are typically a conglomeration of the various sub-networks within an organization. These networks are large and geographically dispersed. They consist of many legacy and modern devices; some that are critical to the network operation itself while others are essential for the services provided. Configuring, managing, and monitoring enterprise networks are a monumental task that requires the requisite network management tools and management expertise.

Generally, enterprise networks are owned by a single organization, such as IBM, federal government bodies, and financial institutions. These networks exist to provide data and telecommunications services to employees, customers, and suppliers. Services can include [28]:

- File and data storage
- Print
- Email

- Access to shared applications
- Internet access
- Intranet
- Extranet
- E-commerce
- Dial tone
- International desk-to-desk dialing (using voice-over-TDM or voice-over-IP)
- Video
- LAN and virtual LAN (VLAN)—often heavily over-engineered (more bandwidth than necessary) to avoid congestion
- Corporate WAN—can be used for data and also voice-over-IP
- Virtual private network (VPN)—can be used for securely joining multiple sites and remote workers and replacing expensive leased lines
- Disaster recovery—maintaining network service after some cataclysmic event

Enterprise networks achieve these and other services by deploying a wide variety of different technologies and systems.

Figure 1 depicts a typical simplified enterprise network. As can be seen, an enterprise network encompasses several functional services such as voice, message, network storage, and application services. Each of the various services supports a number of user specific functions such as email, desk phones, internet access, etc. The connected boxes in the figure provide access to the services. Large networks such as this can serve large geographically distributed corporate users that span over hundreds of remote branch offices [28].

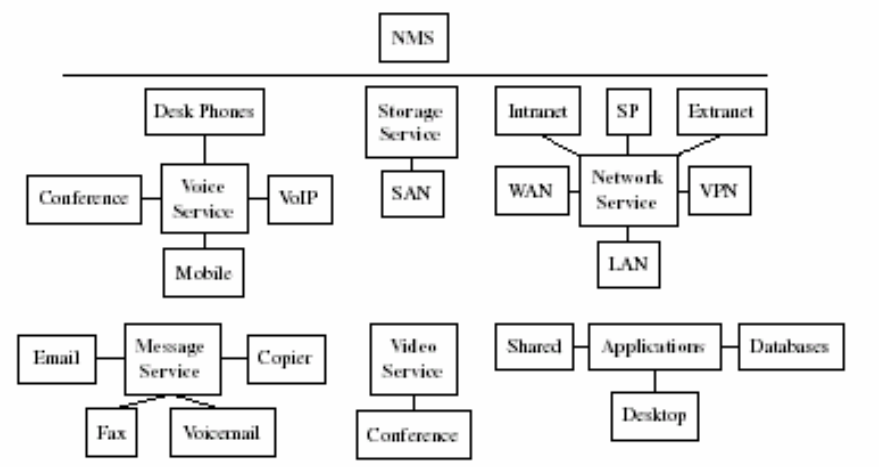

Figure 1. Enterprise network functional components

Enterprise data flows can become very complex once extranets and e-commerce are employed. Extranets are parts of intranets that are extended to organizations external to the enterprise, such as software contractors. E-commerce allows for secure financial transactions between external customers and a given organization. The data flows in the latter case feed into various systems, such as finance, stock control, and manufacturing.

It is apparent that supporting a vast enterprise network across functional systems in a heterogeneous environment call for a powerful underlying network. Following are some general features of enterprise networks [28]:

- They incorporate a wide range of multi-vendor devices, such as routers, switches, exchanges, PCs, servers, printers, terminal servers, digital cross-connects, multiplexers, storage devices, Voice over IP (VoIP) telephones, servers, and firewalls.

- Network elements (NEs) can incorporate other intelligent devices, such as PCs with network interface cards (NICs) and possibly modems. Likewise, desk phones can contain computer-telephony integration (CTI) hardware for applications like call centers and e-commerce bureaus.

- Individual NEs provide a variety of different shared services; for example, a legacy PABX or a soft switch provides basic telephony and can form the foundation of a call center. In this way, a base system is leveraged to provide another system or service.

- Backup and restore of NE firmware are important for rolling out new network services.

- Specialized servers are deployed to provide advanced services such as Storage Area Networks (SANs).
- Many users are supported simultaneously.
- The overall network services, such as email and video/audio conferencing, are used by employees of the organization as essential business process components.

The features and complexities in enterprise networks described above exemplify the massively intertwined networks and services that exist in within the DoD and the military services. For example, within the DoD the Defense Information Systems Network (DISN) is the key wide-area communications component that aggregates a multitude of sub-networks across the enterprise. Some of the DISN networks are listed below:

- Defense Red Switch Network for classified voice conferencing
- Secret Internet Protocol Router Network (SIPRNET)
- Non-Secure Internet Protocol Router Network (NIPRNET)
- Enhanced Mobile Satellite Services (EMSS)
- DISN Voice Communication Systems (DSCS)
- Defense Switched Network (DSN-voice traffic)
- Defense Message System (DMS)

There are also a host of service independent enterprise networks that connect to the DISN for wide-area transport and support. For example, in the Army you have the Army Enterprise Infostructure (AEI) that is under development, the Warfighter Information Network – Tactical (WIN-T) and the Common User Installation Transport Network. Another example is the Navy's Navy Marine Corps Intranet (NMCI) initiative.

2. Network Management

In the late 1970s and early 1980s, networking in organizations began to thrive. With the growing size of the networks and rising number of network platforms and devices, many disparate network management solutions worked there way into these organizations. Many organizations experienced a myriad of network management

problems due to issues such as multi-vendor interoperability, lack of expertise, and reliability. There were a few generic tools available for managing networks; the more sophisticated management tools available were typically proprietary which limited there use. Most available management tools were used in an ad-hoc fashion.

As the networks grew in size and proliferated, they became even more complex to manage. Network administrators and operators had to become adept at handling the many ambiguous anomalies that surfaced. The increased complexity of operations created a demand for common, vendor-neutral, interoperable, and integrated solutions. As a result, two standards emerged in the late 1980s: the Common Management Information Protocol (CMIP) published by the International Organization for Standardization (ISO) and the Simple Network Management Protocol (SNMP) published by the Internet Engineering Task Force (IETF).

Network management comprises all the measures necessary to ensure effective and efficient operations of a networked system. This includes the deployment, integration and coordination of the hardware, software, and human elements to monitor, test, poll, configure, analyze, evaluate, and control the network and element resources to meet the real-time, operational performance, and quality of service requirements at a reasonable cost [7].

The goals of NM are to provide the services and applications of a networked system with the desired level of quality and to guarantee availability and rapid, flexible deployment of networked resources [6]. This includes the detection and handling of faults, performance inefficiencies, and security compromises. To accomplish these goals, management applications are designed to do the following [26]:

- Collect real time data from network elements, such as routers, switches, and workstations. For example, they collect the number of packets handled by the given interface of a router.
- Interpret and analyze the data collected. For instance, they may recognize security events, such as repeated illegal attempts to login on a workstation.
- Present this information to authorized network operators, possibly by displaying a map of current traffic.

- Proactively react, in real time, to management problems, possibly by disabling a link that is experiencing faults.

3. Requirements for Network Management

There are a number of requirements that drive an organization to incorporate a NM system. Below William Stallings [8] provides insight on organizational requirements that justify an investment in NM:

- *Controlling corporate strategic assets*: Networks and distributed computing resources are increasingly vital resources for most organizations. Without effective control, these resources do not provide the payback that corporate management requires.

- *Controlling complexity*: The continued growth in the number of network components, end users, interfaces, protocols, and vendors threatens management with loss of control over what is connected to the network and how network resources are used.

- *Improving service*: End users expect the same or improved service as the information and computing resources of the organization grow and distribute.

- *Balancing various needs*: The information and computing resources of an organization must provide a spectrum of end users with various applications at given levels of support, with specific requirements in the areas of performance, availability, and security. The network manager must assign and control resources to balance these various needs.

- *Reducing downtime*: As the network resources of an organization become more important, minimum availability requirements approach 100 percent. In addition to proper redundant design, network management has an indispensable role to play in ensuring high availability of its resources.

- *Controlling costs*: Resources utilization must be monitored and controlled to enable essential end-user needs to be satisfied with reasonable cost.

These requirements are not only critical for the private sector enterprises, but also they are critical for military organizations to carry out their missions. In fact, these requirements are vital enablers during a wartime situation in order for combat units to communicate within an organization, across the services, with coalition forces, commercial supporters, and CONUS sustaining-base organizations. With military

migrating towards network-centric operations and fielding increasing numbers of network dependent weapon systems, the slightest failure in the network could mean disaster on the battlefield. Therefore, NM in the military must be reliable, efficient and effective from the enterprise-level on down.

4. Network Management Functions

The International Organization for Standardization Network Management Forum has divided network management into five functional areas that are recognized and used as a baseline within the industry: Fault Management, Configuration Management, Accounting Management, Performance Management, and Security Management (FCAPS). The FCAPS principles are further elaborated below [9]:

a. Fault Management

Fault management is the process of detecting and correcting network problems, otherwise known as faults. Faults typically manifest themselves as transmission errors or failures in the equipment or interface. Faults result in unexpected downtime, performance degradation and loss of data. Generally, fault conditions need to be resolved as quickly as possible.

b. Configuration Management

The configuration management functions detect and control the state of the network resources. This entails the initialization, modification, and shutdown of a network. Networks are continually adjusted when devices are added, removed, reconfigured, or updated. These changes may be intentional, such as adding a new server to the network, or path related, such as fiber cut between two devices resulting in a rerouted path. If a network is to be turned off, then a graceful shutdown in a prescribed sequence is performed as part of the configuration management process. The process of configuration management involves identifying the network components and their connections, collecting each device's configuration information, and defining the relationship between network components. In order to perform these tasks, the network manager needs topological information about the network, device configuration information, and control of the network component.

c. Accounting Management

Accounting management functions collect and process resource consumption data. This type of management involves monitoring the login and logoff records, and checking the network usage. This is done to determine a user's use of the network for the purposes of allocation of resources and billing for their usage. Additionally, this type of information helps a network manager allocate the right kind of resources to users, as well as plan for network growth.

d. Performance Management

Performance management involves measuring the performance of a network and its resources in terms of utilization, throughput, error rates, and response times. With performance management information, a network manager can reduce or prevent network overcrowding and inaccessibility. This helps provide a more consistent level of service to users on the network, without overtaxing the capacity of devices and links. This form of management looks at the percentage of utilization of devices and error rates to help in improving and balancing the throughput of traffic in all parts of a network. Typically, some devices are more highly utilized than others. Performance monitoring gives qualitative and time relevant information on the health and performance of devices so that underutilized devices are more fully utilized and overtaxed devices are rebalanced.

e. Security Management

Security management deals with ensuring overall security of the network, including protecting sensitive information through the control of access points to that information; for example, blocking unauthorized access to database records.

5. SNMP Architecture and Functions

As mentioned earlier, the CMIP and SNMP standards were manifested as a result of the great demand for a common, vendor-neutral, interoperable, and integrated network management standard. The SNMP model was originally an interim, rudimentary solution to resolve the growing NM problems. The CMIP model was designed to be more robust and provide greater NM capabilities that would eventually replace SNMP. However, this never happened because the CMIP approach was found to be too complex for widespread adoption. The appeal for SNMP was its small size (lightweight) and simplicity (ease of

implementation, installation, and use). SNMP was widely accepted globally and has now become the de facto standard for network management.

This section is intended to give the reader a fundamental understanding of the SNMP architecture and functions in order to understand the problems with framework discussed later. This section is focused on SNMP, as opposed to other standards, because it is the most ubiquitous and widely accepted open standard today. While the SNMP architecture and functionality are given at an abstract level, a more detailed technical review is provided in Appendix I for a better understanding. Additionally, the other most common standards available are also presented in Appendix I.

The SNMP architecture is a centralized hierarchical design based on the client-server paradigm (see Figure 2). The architecture is made of three core components: managers, agents, and the management information base (MIB). The management logic is performed on a central station (the client) called the management entity or network management station (NMS). The NMS is an "umbrella" application that integrates the user interface with many independent management applications called agents (the server). Agents are software processes embedded on each managed device that monitors, controls, and collects data from the devices. The management data is stored in a database that resides on managed devices where the agents can retrieve it. This database is known as the management information base or MIB. MIBs are organized as static directory trees with managed data stored at the tree leaves [27]. The tree leaves are called MIB objects.

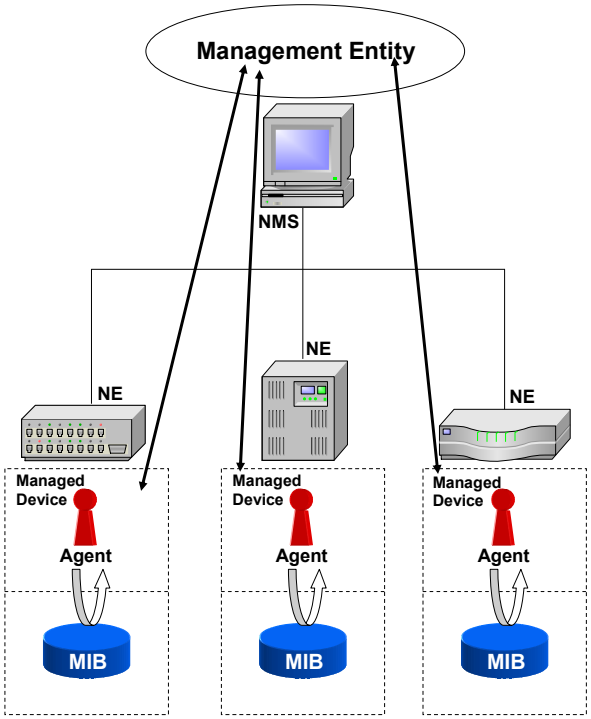

Figure 2. SNMP Architecture

There are a number of vendors that produce network management software and tools on the market. The product capabilities vary in range in terms of the level of management capability, the breadth of management features and functionality. Many products are modular, allowing customers to buy only the necessary features that meet their NM needs. The modules collectively make up a suite that is referred to as "frameworks" in the industry. The more narrowly focused products are known as "point" solutions. Frameworks tend to aggregate the data from point solutions to provide insights on the enterprise network as a whole [34]. There are few giant developers that sell enterprise level suites that are capable of centrally managing networks at the enterprise level. The most recognized enterprise NM manufactures today are: Hewlett-Packard, IBM, Computer Associates, BMC Software, and Aprisma.

The SNMPv1 design consists of a single central NMS, as shown in Fig 2. However, the SNMPv2 design introduced the concept of intermediate network management stations. This design made it possible to decentralize the management burden by sharing the processing load with more than one management station (see Fig

3). The intermediate NMSs are capable of sharing information with one another and with a central NMS that aggregates all the management information for display on a single user interface.

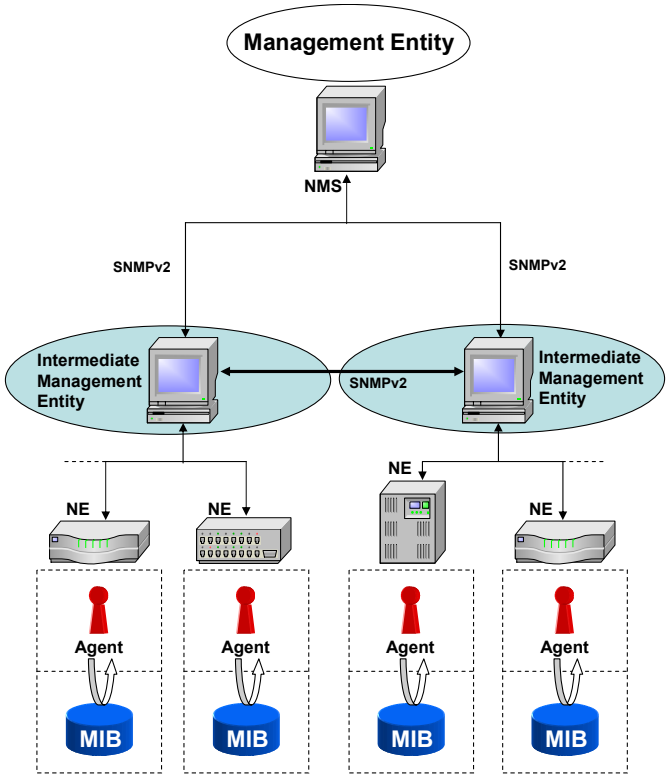

Figure 3. SNMPv2 architecture

SNMP is a polling-oriented protocol that uses a fetch-store paradigm and trap paradigm. Fetch is initiated by the NMS to retrieve values from agents and monitor internal data values and data structures within the MIB. Store is initiated by the NMS to change values on agents and to modify and control data values and data structures within the MIB; it is also use to control behavior of a NE. Trap is initiated by an agent to asynchronously report alarm conditions to the NMS when an unexpected event occurs on a NE.

The SNMP protocol defines exactly how a NMS communicates with an agent. It specifies the message formats, called Protocol Data Units (PDUs), that are used by the manager and agent for requests and responses. It also defines the exact meaning of the request and responses. The protocol primarily uses UDP for transportation to keep the

communications simple and efficient, however it can use other transport protocols such as TCP or HTTP. Instead of defining a large set of commands, the protocol uses the fetch-store paradigm discussed earlier. The protocol defines the syntax for nine management messages: get, get-next, get-bulk, set, get-response, trap, notification, inform, and report.

The SNMP paradigm establishes a "control loop" that involves the collection of monitoring data at the NE, human interpretation and analysis of the computation at the NMS, and the invocation of corrective actions at the NE [27]. Figure 4 shows how the control loop stretches from the managed device across the network to the central NMS. These control loops are subject to failure when the network experiences problems and outages.

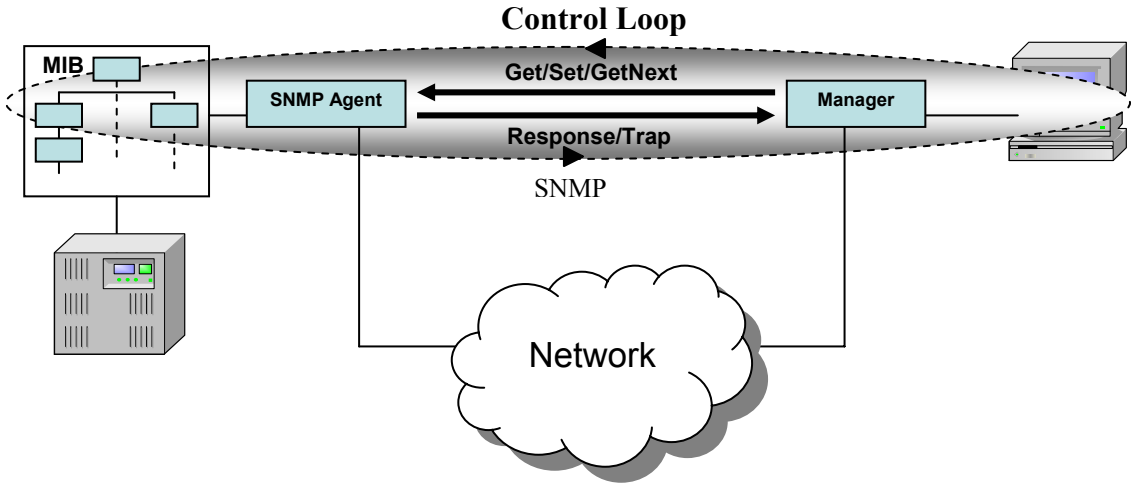

Figure 4. SNMP Control Loop

B. NETWORK MANAGEMENT PROBLEMS AND CHALLENGES

The dimensions and complexities of today's large networks are outstripping the capabilities to manage them in an efficient and cost-effective manner. With the rapid pace of technology advancements, networks have constantly grown in size and consist of a variety of heterogeneous devices and legacy systems. These grand networks have a large number of nodes interconnected by heterogeneous transmission media (e.g. wired and wireless) and operate at accelerated speeds. Managing such networks has increasingly become more difficult, requiring a multitude of tools to centrally manage the

various open and proprietary network components. Often you will find a variety of multi-vendor network management platforms and tools being used to manage a single enterprise network. This is inadequate for the commercial world, as well as for the emerging high-tech military, and has not gone without notice.

There is a great deal of research and study ongoing seeking optimal network management solutions for today's sophisticated network schemes. Many researches view the SNMP and the other traditional protocols as primitive and inadequate to meet the demands of today's and future large-scale networks. This section examines, discusses, and summarizes the various findings under study regarding the limitations and shortcomings of the traditional models most commonly in use today.

1. Shortcomings of Centralized Management Approaches

The centralized nature of client-server network management paradigms, such as SNMP and CMIP, is a major limitation of traditional architectures. German S. Goldszmidt conducted an elaborate study for his dissertation on "Management by Delegation" that chronicles the many shortcomings of the traditional models, with an emphasis on SNMP because of its ubiquity. He points out that the client-server model is too rigid, and thus, hinders the development of effective management systems [27]:

> The implementations of these processes are statically compiled and linked. A client process in a manager role can only invoke a fixed set of predefined services. This set cannot be modified or expanded without the recompilation, reinstallation, and reinstantiation of the server process.

The SNMP framework, for instance, was written based on the assumption that network devices had limited computing resources and therefore had to rely on a central management entity to perform intelligent processing. The next several paragraphs summarize many of the key issues about the problems with centralized management.

a. Scalability Issues

In their study, Puliafito and Tomarchio [11] state that "the rapid expansion of networks has caused scalability problems in managing a larger number of nodes…the larger number of nodes requires increased polling over the network, and causes an increase in network traffic and bandwidth." Goldszmidt further points out that each of these node interactions involves retrieving and analyzing MIB data, which demonstrate

two characteristics: "(1) it concentrates most processing in to the manager's host computer, and (2) it entails a high degree of communications involving the manager's host." [27]

Concentrating the bulk of processing on a central management system is processor intensive and introduces implicit limitations that are detrimental to the network management control. The NMS conducts all of the data computation and presentation that requires high data access and processing rates that do not scale up for large and complex networks. "There is a limit on the maximum number of variables that can be polled and the frequency of polling." [27] Depending on the processing power available, the NMS is limited by the processor capabilities. This can limit the NMS on how many managed objects can be polled, and how often they can be polled. Thus, the more devices to be managed, the greater the limitation imposed. Finally, the centralized model creates a single point failure at the NMS that could eliminate communication and management interaction with all the virtually connected managed devices [27]. Goldszmidt presents an example about automating the management of routers that clarifies the potential scalability dilemma:

> Consider an organization that wishes to automate the management of its routers. That is, the organization wants to deploy programs that (1) monitor the operations of the routers, (2) analyze their behaviors and (3) invoke appropriate control functions. For example, suppose one wishes to deploy programs that monitor routing tables to detect routing problems and invoke appropriate handlers. When the network is large and fast, remote polling of large routing tables may consume significant bandwidth resources. The NOC hosts may be unable to detect and handle remote problems sufficiently fast. Centralization thus seriously limits the scalability of a network management system [27].

An argument can be made that SNMPv2 mitigates this problem by enabling shared management processing. However, even with SNMPv2 the limitations persist, but with a lesser impact. The SNMPv2 intermediary managers serve as single point of failure for the devices that they manage, and they can be overwhelmed depending on the number of devices managed and the magnitude of interaction with the managed agents. Bandwidth is still wasted with the static constant polling of the managed agents that in many cases is not necessary. Using distributed management

servers can be costly for large networks that have hundreds of devices to be managed. The more devices there are to be managed, the greater the cost for additional intermediate managers to control them.

Scalability in a large enterprise network is certainly a paramount concern when considering the rates at which operators can handle the volume of data and alerts. Network administrators can easily become overwhelmed by a flood of messages that may or may not be a cause for great concern. This is more of a problem as the network grows. Ideally, it is more efficient for programmed software to interpret and distill the information, and react autonomously to alleviate the human operator. However, the limited capabilities of today's centralized protocols establish significant barriers [27].

b. Reliability Issues

It is somewhat of a paradox to employ a centralized model to manage and control a network that experiences network congestion, delays, and failures that renders network management and control by the NMS helpless. A centralized model is unreliable. It is at the mercy of the network. Managed devices cannot accomplish recovery without instructions from the NMS. This is due to the agent's lack of intelligence, although RMON, which is incompatible with SNMP, provides some degree of mitigation. RMON issues are discussed later.

There is a greater potential for reliability issues in complex, large-scale networks. "During times of failure, centralized management tends to increase the rate of data access at a time when the network is least capable of handling them." [27] "The larger number of nodes requires increased polling over the network…this becomes even more of a problem during high congestion periods when there is a need for management actions." [11] This is yet another paradox in that during times of failure, when management action is vital, the NMS exacerbates the problem by attempting to increase the interaction with the managed devices, potentially causing a communications "bottleneck."

2. Other Shortcomings

a. SNMP Deficiencies

Goldszmidt provides an articulate summary of the more technical deficiencies of SNMP that are worth presenting here [27]:

SNMP polling introduces significant delays in retrieving management data to the platform. These delays are due to: (1) transient conditions, e.g., network contention or congestion, (2) configuration problems, e.g., the routing distance between the devices and the platform, and (3) the protocol design, e.g., the need for ASN.1 parsing of management PDUs in both communication endpoints (device and platform). High frequency polling introduces large bandwidth overhead. Slow polling will miss transient spikes (errors, load, etc) as it will average it over long periods of time.

SNMP-agent implementations introduce big timing errors in the observations of real devices, which produce outdated, and potentially erroneous, data in the agent's MIBs. Typically, MIB tables change while a management application is retrieving or examining them. Inaccuracies like these often lead to erroneous computations.

The following list outlines several of the problems associated with SNMP implementations:

- SNMP uses the network to transmit information about network measurements. Thus, it introduces an intrinsic disturbance.
- When a device is loaded, its SNMP-agent is scheduled with relatively lower priority, and thus queries to it will often be delayed.
- Event report traps are unacknowledged and an unreliable protocol (UDP) is used to deliver them. Thus, an agent cannot be sure that a trap has reached its destination;
- The MIB model does not support queries based on object values or types. Thus, applications can not filter MIB data at its source, and must retrieve large amounts of MIB data.
- Many implementations of SNMP-agents are erroneous and return wrong data.

b. Heterogeneity and Convergence Difficulties

The management of heterogeneous networks requires the capabilities to account for events occurring on different time scales and the capability to aggregate different types of data. In their study, Gurer, Lakshminarayan, and Sastry [10] found that there is no apparent single NM technology available that has the capability to fulfill the real-time quality of service (QoS) needs of the various applications and network technologies. Voice, video, and data each have different timing requirements that must

be monitored, managed and controlled by the NM system. This entails the simultaneous collection of different types of data based on the technology, analysis, and the appropriate decision being made for that technology. Standards such as SNMP and CMIP are not sophisticated enough to conduct this type of collection, analysis, and proactive decision-making. These standards only provide simple data gathering and reporting.

A Lucent Technologies, Inc., research group [17] point out that the existence of multiple standards based on the different types of networks and technologies (i.e. voice, video, and data standards) is another NM impediment. These various types of networks and technologies have different specifications and management requirements that caused the creation of competing standards such as SNMP, CMIP and telecommunications management network (TMN). The competing standards have lead to different communities adopting different standards. The data community has generally adopted the SNMP standard, while the telecommunications community mostly adopted CMIP and in some cases the TMN protocol standard.

c. Remote Monitoring Shortcomings

In a research study by Gavalas, Ghanbari, and O'Mahony [13], they state that the RMON distributed management model also presents some considerable NM limitations and issues. Although RMON reduces the amount of bandwidth traffic and processing burden on the NMS, it is still based on a client-server centralized architecture. In other words, the real intelligence and processing still resides in the NMS. Since a single RMON device is required to monitor the traffic of a single network segment, it becomes very costly as the number of segments increase. Due to the fact that the RMON probe can only be set or modified during configuration makes it very inflexible for dynamic changes during runtime. Finally, RMON is limited to providing only traffic-oriented statistics as opposed to node-oriented statistics.

d. Market Findings

Trying to find a comprehensive study on best enterprise network management products on the market is hard to come by. The reason for this is perhaps there is no true enterprise network management solution that incorporates the full spectrum of NM functionality. This is consistent with what the many researchers have found regarding the NM shortcomings. This is essentially the reason why the military

cannot simply buy a commercial NM product off-the-shelf to fulfill their enterprise network management requirements. Consider the following comments made in a recent article addressing the limitations of SNMP to comprehensively achieve FCAPS [35]:

> …These limitations partly revolve around what can--or can't--be done with SNMP. As a standard, SNMP is widely installed and leveraged by network-management vendors but is primarily limited to addressing the fault and performance portions of FCAPS. Frankly, even expensive network-management products are hamstrung when they rely solely on SNMP data, which is why many have proprietary agents.

Iosif G. Ghetie [12] conducted a market study on major marketed NM products that revealed a lack of cooperation and integration between NM applications. Here are some of the noteworthy findings from the study that still prevails today:

- None of the products are able to fully cover all the network management areas. Their main focus is on network monitoring and event reporting.
- None of the products are able to easily manage heterogeneous networks.
- Scaling is difficult due to the inadequacy of the application development tools.
- The systems are very resource consuming (e.g. SNMP polling).
- All are expensive.

3. Final Point

The problems and issues discussed above regarding the prevalent network management models will impose limiting factors on the GIG and the Army's enterprise management initiative. This chapter set out to define the current NM standards of today and clearly exposed the inherent weaknesses that will limit the military's efforts. Without more flexible, adaptable, and scalable protocol standards the GIG vision probably will not be truly realized to the fullest extent. The shortcomings of the modern day protocol standards can result in the failure of reaching the full spectrum dominance objective, which translates to handicapping the warfighters and possibly compromising battlefield successes. Emerging intelligent-agent-based technologies and multi-agent systems offers robust capabilities that seemingly provide the necessary characteristics required to manage enterprise networks as vast as the proposed GIG and Army AEI.

III. INTELLIGENT AGENTS AND MULTI-AGENT SYSTEMS

A. THE AGENT WORLD

The complexities of network management are growing beyond the capabilities of the current centralized and distributed NM systems. These classical approaches lack the adaptability, flexibility, reliability, and scalability that are necessary to manage the vast inherent in today's and future enterprise infrastructures. Heterogeneous enterprise network environments are laced with legacy systems, proprietary solutions, and disparate open standard NM protocols. Even modern day technologies are lacking. "Emerging technologies, like CORBA for example, do not seem to be able to solve problems of complexity, cost and scalability" [14].

Intelligent Agent (IA) technologies are on the horizon and appear to a most promising solution to resolve many of the enterprise NM pitfalls. This section gives and overview of intelligent-agent-based technologies and Multi-agent Systems (MAS).

1. What is an Agent?

There exist unsettling debates as to the definition of what an agent really is in the agent communities. There are different schools of thought from the various disciplines such as the artificial intelligence (AI) or computer science communities. Certain attributes may be of more importance in one discipline than in another. For example, in some disciplines, the ability for agents to learn from their experiences is of paramount importance, while in others it is not. However, most disciplines commonly agree that autonomy is central to the notion of agency [16].

Fundamentally, an agent, in the context of software, is essentially a self-contained software program module that is programmed to carry out certain actions on behalf of a human user or other software entity in a certain software environment. The agent can perform such things as searching for information, negotiating services, executing specified tasks, or collaborating with other agents. These actions are conducted in an autonomous fashion that requires little or no human intervention.

A highly recognized definition of an agent comes from Micheal Wooldridge and N. R. Jenning [15]: An agent is a computer system that is situated in some environment,

and that is capable of autonomous action in this environment in order to meet its design objectives. Autonomous in this sense means that agents are able to act without the intervention of humans or other systems: they have control both over their own internal state, and over their behavior. "Another common view of an agent is that of an active object or bounded process with the ability to perceive, reason, and act." [18] Keep in mind that the definitions above refer to the general concept of "agents" and not necessarily "intelligent agents," which is defined later. An unintelligent agent is distinguished from an intelligent agent based on the agent's properties.

Looking back at SNMP, recall that the managed device contains a software entity called an agent. In contrast, the SNMP defined agent does not qualify as an agent as expressed in the context of the preceding paragraphs. The SNMP agent has absolutely no degree of autonomy or intelligence. The SNMP framework was written with the assumption that network devices have limited computing resources available, and therefore must rely on the NMS for instructions and computational processing. Even considering the SNMP Trap, the agent is limited to a fixed set of predefined thresholds that are statically implemented. This exemplifies the wide abuse of the term "agent."

2. Agent Properties

Considering the different perspectives on the definition of an agent, it is useful to look at the varying properties (capabilities) that an agent can assume. Cheikhrouhou, et al. [14], provide a list of several properties that commonly characterize agents:

- **Autonomy.** Self-government, independence: Branch managers have full autonomy in their own areas (Oxford Advanced Learner's Dictionary). The agent decides himself when and under which condition he will perform what actions. An autonomous agent is a system situated within and as a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda so as to effect what it senses in the future.
- **Communication.** One of the key properties of agents is the ability to speak with a peer, with a human (Interface Agents), or with a device. The following communications between agents, called languages, are often used:
    - Blackboard: Agents read and write messages in a shared location, called a blackboard.

- KQML: Knowledge Query and Manipulation Language is a language and protocol for exchanging information and knowledge using what is known as "performatives" (discussed later).
- KIF: Knowledge Interchange Format.
- COOL: structured conversation, KQML-based, which is used for the coordination of agents.

- Collaboration/Cooperation. Agents are collaborative when they are able to work together. The agent is able to communicate and negotiate with others; it is deliberative and may coordinate its actions with others. Collaborative agents are particularly useful when a task involves several systems on the network. Negotiation is the main issue for collaborative agents. While coordination can occur without collaboration, collaboration needs negotiation.
- **Deliberation.** Know rules, and apply them without waiting for instructions. Wooldrige and Jennings define a deliberative agent as "one that contains an explicitly represented, symbolic model of the world, and in which decisions (...) are made via logical (or at least pseudo-logical) reasoning, based on pattern matching and symbolic manipulation."
- **Mobility.** Since the arrival of Java portability, a number of mobile agent models have surfaced. But, there are different kinds of mobility defined:
  - The mobility that allows the agent to move from one system to a similar one.
  - The mobility that allows the agent to move to another different system.
  - The mobility that allows agents to suspend their action on one system, move to another and go on.
  - The mobility that allows the agent to move itself, rather than being transported.
  - The mobility that is a duplication of the agent to another system (cloning).
  - The mobility that allows agents to carry its knowledge to another system.

  Generally, mobility turns out to be a mixture of these definitions. One of the main issues surrounding mobility is the potential security weakness of mobile agents.
- **Learning.** Learning is the ability of an agent to acquire knowledge and use it to modify its behavior. Despite the fact that learning is an important factor of intelligence, few agents are able to learn. Most often they have

fixed (pre-compiled) rules and knowledge bases. The objective of learning is for the agent to perform new tasks dynamically without being stopped. Different ways of learning are studied and experimented:

  - Generalization: you observe your environment and deduce rules.
  - Instruction: you obtain knowledge and rules from others (transfer).

- **Pro-activeness.** Pro-active actions are intended to cause changes, rather than just reacting to change. Pro-active agents generally follow plans, or at least execute rules when the environment reaches a known threshold. Sometimes pro-active is used with the same meaning as deliberative, but an agent may be pro-active, because it has been requested to perform pro-active tasks, as opposed to deliberative agents, who decide themselves to be pro-active.
- **Reactivity.** Do something when an event occurs.
- **Security.** Be able to discriminate friends from enemies and contaminated elements.
- **Planning.** The agent organizes by priorities the actions to perform during its life. For many researchers planning is one of the most important properties for an intelligent agent to possess. Planning is used by deliberative and pro-active agents according to their knowledge of the environment and the possible actions that they can apply to it.
- **Delegation.** An agent may ask another agent to perform one of his goals or tasks. This capacity is very important for balancing resources.

It is important to note that the descriptions above are not all inclusive. The various disciplines would defend their position as to what descriptions constitute their ideas of an agent. But, as can be seen, the properties of agents are quite extensive and sophisticated. Appling intelligent agents to the NM arena are a matter of incorporating the necessary properties to deal with the needs for the NM complexities involved in future enterprise networks.

3. What is an Intelligent Agent?

Just as with the debate over the general agent definition, there is no universally agreed upon definition of an intelligent agent. Considering the previously defined definition of an agent and the various properties, an intelligent agent assumes all the characteristics of the agent definition, but, also assumes a mixture of the properties that are determined based on the IA design or on one of the several beliefs of a particular community. For example, Wooldridge [16] defines an IA as one that is capable of

flexible autonomous action in order to meet its design objective, where flexibility means three things:

- **reactivity**: intelligent agents are able to perceive their environment, and respond in a timely fashion to changes that occur in it in order to satisfy their design objectives;
- **pro-activeness**: intelligent agents are able to exhibit goal-directed behavior by taking the initiative in order to satisfy their design objectives;
- **social ability**: intelligent agents are capable of interacting with other agents (and possibly humans) in order to satisfy their design objectives.

Different kinds of intelligent agents have different subsets of properties. Thus, the various disciplines have different views of an intelligent agent. This pretty much sums up the general IA context:

> Most agree though that to be intelligent, agents must include the ability to operate in real-time and communicate using natural language. Along with this, they must be able to learn from their environment and be capable of adaptive goal-oriented behavior. In other words, intelligent agents need to work together on a user-specified problem when told to do so and must be able to do this successfully in a dynamic environment. Importantly, the agent must communicate to the user, in a language he or she understands, that the task has been successfully completed or that it has been otherwise terminated.

4. Multi-Agent Systems

The real benefit of agent technologies is leveraging the capabilities of multiple agents that have different functions and have the ability interact with other agents (or humans) to solve problems and execute tasks. A multi-agent system (MAS) is a subset of Distributed Artificial Intelligence (DIA). DIA is concerned with problem-solving where agents solve tasks in a collaborative manner, in a distributed environment.

A MAS is a platform composed of multiple agents that interact to solve problems beyond their individual capabilities or knowledge. Interaction [18] is everything that occurs between agents (agent-agent interaction) and their environment (agent-environment interaction). Agents can interact directly via verbal communication (e.g. by providing information in which other agents are interested or which confuses other

agents) or indirectly via their environment (e.g. by observing one another or by carrying out an action that modifies the environmental state).

The idea of MASs is to distribute functionality and intelligence in a decentralized manner where agents can dynamically solve exclusive problems only when necessary; providing flexibility, adaptability, and efficiency. The strength of MASs is agents collaborating and cooperating to collectively solve large, complex problems that are beyond the capabilities of any single agent. This point counters the architectural nature of the traditional NM protocols where the agent-NMS interaction is inflexible and rigidly defined.

An MAS has the following advantages over a single agent or centralized approach [46]:

- An MAS distributes computational resources and capabilities across a network of interconnected agents. Whereas a centralized system may be plagued by resource limitations, performance bottlenecks, or critical failures, an MAS is decentralized and thus does not suffer from the "single point of failure" problem associated with centralized systems.
- An MAS allows for the interconnection and interoperation of multiple existing legacy systems. By building an agent wrapper around such systems, they can be incorporated into an agent society.
- An MAS models problems in terms of autonomous interacting component-agents, which is proving to be a more natural way of representing task allocation, team planning, user preferences, open environments, and so on.
- An MAS efficiently retrieves, filters, and globally coordinates information from sources that are spatially distributed.
- An MAS provides solutions in situations where expertise is spatially and temporally distributed.
- An MAS enhances overall system performance, specifically along the dimensions of computational efficiency, reliability, extensibility, robustness, maintainability, responsiveness, flexibility, and reuse.

Wooldridge [16] discusses two contrasting patterns of coordination when agents interact – cooperation and competition. In cooperation several agents work together by drawing on their knowledge and capabilities to achieve a goal. These agents fail or succeed together because they try to accomplish collectively what individual agents

cannot. With competition, agents' goals are conflicting so they work against each other. Competitive agents try to maximize their own benefit at the expense of other agents, thus, the success of one implies the failure of others.

5. Agent Architecture

The forgoing discussion of agents just merely provides a cursory overview of the principles of agents. However, the theory and architectural makeup of agents is much more complex and ambiguous. This difficult subject matter is beyond the scope of this study, however, it is important to understand that there is much more involved in the architecture of an agent. There are several approaches for designing and developing agents. An agent architecture [18] is:

> a particular methodology for building agents. More generally, the term is used to denote a particular arrangement of data structures, algorithms, and control flows, which an agent uses in order to decide what to do. Agent architectures can be characterized by the nature of their decision making. Example types of agent architectures include logical-based architectures (in which decision making is achieved via logical deduction), reactive architectures (in which decision making is achieved via simple mapping from perception to action), belief-desire-intention architectures (in which decisions making is viewed as practical reasoning of the type that we perform every day in furtherance of our goals), and layered architectures (in which decision making is realized via the interaction of a number of task accomplishing layers).

6. Agent Communication

Once agents are created, they need a mechanism for communicating between agents, application, and human users. Agent communication [17] is accomplished with three components: ontology, content language, and agent communication language (ACL). Agents use ontologies to limit the scope of their interactions and focus on a specific world of understanding. The content language is used for information encoding through statements about the domain, which combine terms from the corresponding ontology into meaningful sentences. The ACL acts as a formalism for exchanging messages.

An example of one of the most often used ACLs for communication exchange is the Knowledge Query and Manipulation Language (KQML) [14] (the Foundation for Intelligent Physical Agents (FIPA) is another). It was developed under the Defense

Advanced Research Projects Agency (DARPA) Knowledge Sharing Initiative. KQML is a protocol for exchanging information and knowledge.

KQML is based on speech act theory [14], which is founded on the idea that with language you not only make statements, but also perform actions, such as requests, suggestions, commitments, and replies. For example, when you request something you do not just report on a request, but you actually effect the request. These performance actions, or requests, are called performatives. Example performative verbs include promise, report, convince, insist, tell, request, and demand.

There are three main aspects of speech act [19]:

> The *locution* refers to the lowest level of the speech act, namely, the string that is transmitted. The *illocution* refers to the intrinsic meaning of the speech act. The *perlocution* refers to the possible effects of the speech act on the recipients. The locution can be varied and the perlocutions depend on the receipent. However, the illocution tells us the meaning that is conveyed.

KQML divides communication into illocutionary categories [14]: assertives (statements of fact), directives (commands or requests), declaratives (announcements of actions taken), commisives (commitments) and expressives (expressions of emotion). For example, KQML uses performatives such as tell, which asserts a belief; deny, which asserts a disbelief; ask-if, which requests information; error, which asserts a message was read incorrectly; and sorry, which asserts that a reply or task cannot be undertaken.

The fundamental objective of agent interaction is to separate the semantics of protocol communication protocol from the semantics of the enclosed message. While the semantics of the communication protocol must be domain independent, the semantics of the enclosed message may depend on the domain. The communication protocol must be universally shared by all agents. With KQML, all the information for understanding the content of the message is encapsulated in the communication itself. The KQML protocol has this basic structure [19]:

```
(KQML-perfomative
                    :sender      <word>
                    :receiver    <word>
                    :language    <word>
```

:ontology <word>
:content <expression>
…)

The first line of this format indicates the KQML performative (e.g. “tell”). As discussed above, since the KQML is based on speech act performatives, the semantics of KQML performatives is domain independent. Think of the KQML performative as a header that wraps (encapsulates) the message information for exchange. The wrapped message may be domain dependent in order for the recipient to understand it. The :sender and :receiver fields identify the sender and the receiver of the message. The :language field identifies the language in which the message is expressed. The :ontology field denotes the ontology that contains the vocabulary necessary for collaboration and comprehension. Finally, the content field is the message or instruction itself. Note that there are other fields, such as :in-reply-to, that can also be embedded in the structure [19]. Here is an example [10]:

(ask-if
:sender Problem_Detection_Agent
:receiver Control_Agent
:language C++
:ontology Action_Request
:in-reply-to NA
:content “Initiate(Traceroute (Node_36))”)

In this example, the Problem Detection Agent is requesting that the Control Agent send a traceroute message to Node_36 and return the message to the Problem Control Agent.

In KQML, agents communicate either synchronously or asynchronously. The difference is that for synchronous communication the sending agent waits for a reply, whereas with asynchronous communication it continues with its reasoning or acting until it receives a reply. There are many other dynamics that are within the KQML protocol that enable agent communication [19].

B. AN ARGUMENT FOR INTELLIGENT AGENTS IN NETWORK MANAGEMENT

A group of researchers at Lucent Technologies, Inc. [17] created a agent-based platform call LucINA (Lucent Intelligent Agent Network) to test and evaluate intelligent-agent-based NM and other agent related technologies. Based on their research and experimentation they have made a compelling argument for intelligent-agent-based technologies as a premier solution for NM of large heterogeneous enterprise networks. The following paragraphs synopsize their findings.

1. Dynamism

The growing incompatibility of multi-vendor equipment and dynamic changes in network topologies has caused increased complexity and dramatic structural changes in network architectures. The Lucent group found that IA systems are better suited for managing such dynamically changing environments. They concluded that agent-based systems handle dynamism in a natural way because these types of platforms provide for controlled agent life cycle. Agents can be added to the system at will, via registration mechanisms. Agents can advertise itself and its services, making network discovery automatic. The meta-level facilities are used to dynamically discover supported communication parameters such as language, protocol, and ontology.

Traditional and other management frameworks are inflexible and limited compared to agent-based systems. Frameworks based on SNMP and CMIP require software recompilation to handle changes, and offline time is required to activate new or modified software. CORBA-based systems implemented with solely static invocations suffer from the same problems. The use of dynamic invocation interface (DII) with COBRA presents a time-consuming nuance relative to agent-based systems - it is cumbersome for programmers to code. Web-based solutions are completely reactive, which means that if the NMS doesn't ask for data then it will not be delivered.

2. Multiple Standards

The diverse user requirements for the various network technologies have led to the creation of several different network management standards and proprietary products (this is discussed in Chapter II). At the highest level, agents use a uniform communication means (e.g. KQML) for interaction between heterogeneous agents. Any

other standard can be applied at the content level. No other network management technology has this degree of flexibility.

3. Interoperability

Network management interoperability issues have resulted from the development of dissimilar network management information models and manager-agent communication protocols. In agent-based systems the meta-layer for exchanging communications acts allows for coherent exchange of data at the content level. Agents know exactly what kind of data to expect and what the meaning is. This allows for agent-to-agent understanding in which data are processed automatically without prior arrangements. Ontologies serve to mitigate interoperability issues by structuring data, describing relationships and rules governing the data, as well as processing algorithms. The Lucent research group asserts that ontologies are much easier to standardize than traditional standards because they are relatively smaller in size.

Interoperability among the other NM technologies still suffer from unresolved issues. The traditional NM standards still have many incompatibility issues that can only be resolved by modifying code. CORBA does not have a negotiation mechanism that allows for completely ad hoc use of arbitrary added object. In the case of Web-based solutions, there is no way of implementing XML standardization without implementing a Web server and a Web browser as de facto negotiating agents.

4. Distribution

Current solutions for distributed management, such as RMON, are perhaps not truly distributed due to their inherent centralized nature. CORBA, as a distributed NM solution, suffers from many problems such as information bottlenecks and single points of failure. The advantage of an agent-based system is that it is inherently distributed. The communication language provides for natural collaboration between agents at various levels. This allows for a bottom-up approach to problem solving and utilization of available services at any level and stage. The mobility of agents is also an advantage. Mobility and intelligent distributed processing can provide efficiencies such as lower bandwidth requirements for network management.

5. Security

As enterprise networks continue to grow in complexity, the stakes for security becomes increasingly more critical. Because of the inherent vulnerabilities of software security, there is no clear advantage that IA technologies have over other technologies. The agent-based framework does allow for security schemes at several layers. This framework is convenient to enforce security because agents decide at run time whether a received request will be fulfilled. All the other current technology frameworks provide security mechanisms because the industries mandate it within the standards. SNMP and CMIP standards mandate security schemes that have to be implemented at the design and deployment phases. CORBA provides security mechanisms integrated with the platform. Web-based approaches depend on the security implemented by Web browsers and servers.

6. User Experience

As networks continue to expand, more and more users will become involved in some form of network management. Web-based solutions have addressed this by providing user-friendly graphical user interfaces (GUIs) that simplifies NM for the common user. The Lucent group suggests that agent-based systems stand out because they deliver plug-and-play networks. Network elements do not have to be provisioned because the agents residing on them can negotiate the conditions for incorporating them into the network. Similarly, services can be plugged into the network and made available automatically due to negotiation and directory services.

7. Rapid Software Delivery

Today's competitive marketplace elicits rapid design, deployment, and maintenance of software products, as well as controlling the costs of expansions and modifications. Therefore, systems have to be designed, implemented, and deployed quickly. Agent-based systems are very promising in this regard. They are designed with high-level concepts because the platform handles most low-level technicalities such as message passing through method invocation. Thus, systems can be designed by network management experts as opposed to programmers. In contrast, architectures built on top of SNMP and CMIP standards are the most expensive to design, implement, deploy, and

maintain. Modifications require plenty of changes to the configuration data as well as off-line time. To accommodate new advances in standardization, software components have to be recompiled and relinked. Similar criticism is applicable to Web-based approaches. CORBA-based systems can take advantage of well-established object-oriented analytical and design methodologies and development tools. Systems that take full advantage of these capabilities are flexible and easy to upgrade.

8. Cost

Cost is a major factor for any organization when deciding to procure a software system (especially when buying enterprise NM software). So much so that cost cutting has led to the evolution of management tools from proprietary and semi-proprietary solutions based on specialized platforms to standards-based solutions. The Lucent group argues that the efficiencies (as discussed throughout) that agent-based systems bring will result in lower expenses for NM. The Lucent group underscores that the value of the use of artificial intelligence techniques in providing human-like behavior will result in substantial savings in direct costs due to the decreased requirement for human operators. This holds true for the other technologies, but the capabilities of agent-based systems provide a much greater value.

The lucent findings discussed are consistent with the shortcomings of the classical protocols presented in Chapter II. The evaluation findings provide comparative proof and demonstrate the potential for intelligent-agent-based technologies as an overall better solution for enterprise NM, relative to SNMP and the other common protocols.

C. MULTI-AGENT SYSTEM MODELS

There are quite a few multi-agent models that are under development, active and commercialized. This section provides the reader some insight to the nature of the different MAS approaches. The models presented here are mostly derived from a research project [37] that conducted an indebt analysis of MAS platforms that were most suitable for the network management domain. Since the project publication, some of the platforms have matured, some have dissolved, and some have been commercialized. However, the overall objective here is to provide a sense of the various approaches that can be undertaken. There are other platforms that are not mentioned here for several reasons. Some platforms are proprietary and not openly available, many are not suited

for NM, and others are just simply unknown. More interested readers are urged to explore the web and sites such as: www.multiagent.com, www.agents.umbc.edu.

The research project considered several factors for selecting the multi-agent systems discussed below. The following is the list of criteria used [37]:

- **Communication**: What protocol does the system use? Is it flexible? Does the system provide directed communication and/or multicast communication?
- **Programming Language**: Is it a standard language? Easy to use? Compatible with other components? Portable?
- **Flexibility:** Is it easy to adjust the system to a particular application? What are the constraints and requirements?
- **Architecture**: Is the system object oriented? Layer based? Well designed?
- **User Interface**: Can the agents be visualized? How does the user interact with the system?
- **Scalability**: Does the system adapt itself to different situations? Can the system be widely extended?
- **User friendliness**: Is it easy to start using the system? Is the learning curve low?
- **Identification**: How do agents identify one another? Is it a centralized way or through communication?
- **Security**: Are they any security features provided? Are the communications among agents encrypted?
- **Extra features:** Are any extra-features available? Are the agents mobile? Are any coordination constructs available?

The next several paragraphs are summaries from the research project [37] describing the characteristics of the various MAS platforms. It some cases, a synopsis of

the research findings are also presented. The MAS suggested in this study, CoABS, will be described in greater detail in Chapter VI.

1. JAFMAS

Java-based Agent Framework for Multi-Agent Systems (JAFMAS) [38] is a Java-based framework for representing and developing cooperation knowledge and protocols in a multi-agent system. The framework enables the agents to work together and coherently achieve their goals and those of the multi-agent community as a whole. JAFMAS defines a generic methodology for multi-agent application development and provides a set of services that relieves the developer from the effort of programming cooperation mechanism form scratch. It guarantees that essential interoperation, communication and cooperation facilities are available to support agent application developers. JAFMAS is concerned with coordinating intelligent behavior among collection of intelligent agents forming the multi-agent system. Agents should coordinate their knowledge, plans and goals so that they can take actions which results in a joint coherence solution to the problem at hand.

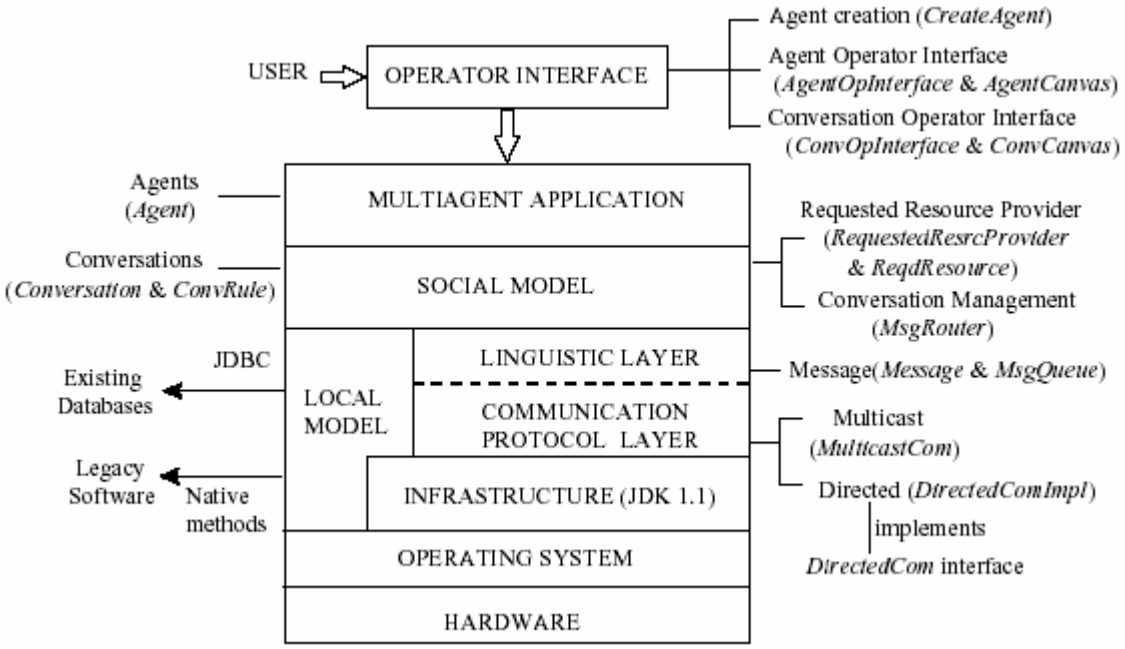

Figure 5. JAFMAS architecture [38]

Figure 5 shows the entire JAFMAS architecture and the classes composing the different layers. JAFMAS provides sixteen main Java classes as shown (name of classes between parenthesis). Those classes provide the essential communication, interaction and coordination mechanisms to application developers by dividing the services provided into distinct layers.

2. JATLite

JATLite (Java Agent Template, Lite) [39] is a package of Java classes and programs that allow users to create quickly new systems of software agents that communicate over the Internet in order to perform a distributed computation. The agents may be newly created software or legacy software "wrapped" with software that generates and receives agent messages as an integration mechanism. In addition to code for creating agents, JATLite provides a robust agent infrastructure, as shown in Figure 6, in which agents register with an agent Message Router (AMR), using a name and password, in order to be able to exchange buffered messages with and transfer files between other agents on the Internet (some of which may be Java applets), and connect/disconnect/reconnect from/to the joint computation. Communication may be asynchronous and intermittent agents are supported. There is no requirement for installation of special software to host agents and no special host is assumed for any agent.

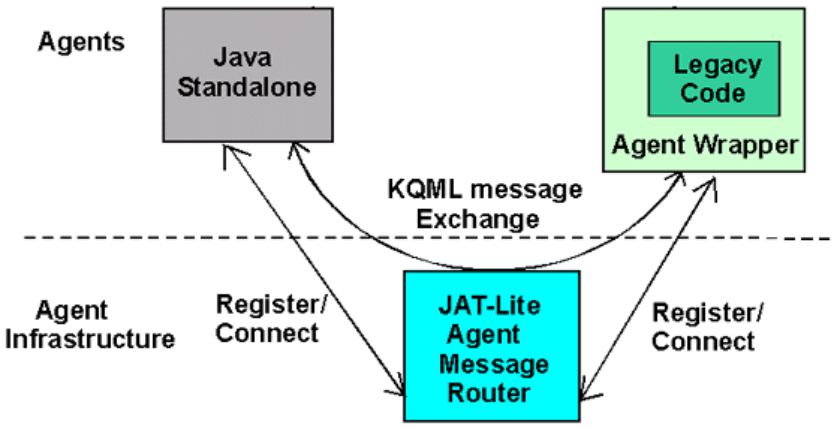

Figure 6. JATLite Agent Message Router [39]

The most important service of JATLite is the Agent Message Router (AMR) that allows agents to fail and recover, to migrate, and to be applet-based. The AMR buffers and forwards messages like an email server. Each agent makes a single socket connection to the AMR - this is the only IP address each agent knows, in addition to its own. The AMR then forwards the message to the correct IP address for the recipient. If the recipient agent is not connected (there is no active socket connection), the message is delivered to the agent when it does connect again. The messages are saved on the AMR until the recipient agent sends a delete signal. This simple idea eliminates lost messages due to temporary agent failure, the necessity for agents to track IP addresses, and the restriction on applet communications, as there now need be only an AMR on the server that spawned the applet in order for it to exchange messages with any other agent connected to the AMR [39].

Although JATLite does provide essential functionality required for building a multi-agent application, it does not define a methodology for specifying the social behavior of agents. Moreover, the concept of the AMR is inherently centralized in nature. All communication must go through the AMR. Each time an agent joins and leaves the system, it has to inform the AMR. This can lead to scalability problems [37].

3. Aglets

Aglets [40] Workbench is a visual environment for building network-based applications that use mobile agents to search, access, and manage corporate data and other information. Aglets are mobile Java programs which may travel and execute in specialized nodes in the network. The Java Aglet Application Programming Interface of the framework defines the methods necessary for Aglet creation, message handling in the network and initialization, dispatching, retraction, deactivation/activation, cloning and disposing of the Aglet.

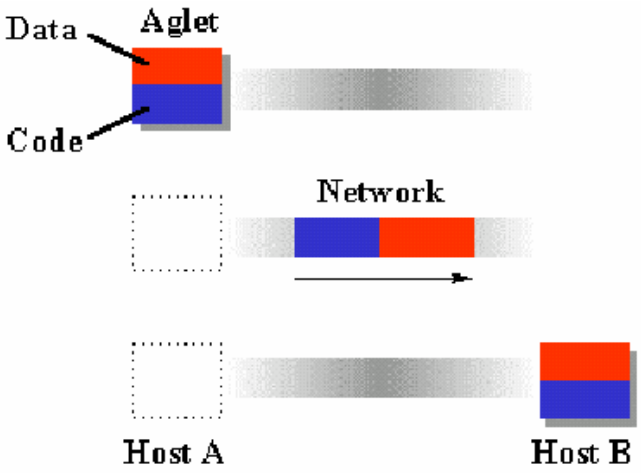

Figure 7. Aglet Serialization through the network [37]

The Aglets workbench includes an Agent Web Launcher named Fiji and a Visual Agent Manager named Tahiti. Fiji is a Java applet based on the Aglets Framework and therefore capable of creating an Aglet and retracting an existing Aglet into a client's web browser. Tahiti uses a unique graphical user interface to monitor and control Aglets executing on a given computer. It also implements a configurable security manager that provides a fairly high degree of security for the hosting computer system and its owner. Although, Aglet is more intended to allow agents to move than a framework for multi-agents, it can be combined with JKQML to allow communication among agents. JKQML was developed to provide a framework and API for constructing Java-based, KQML-speaking software agents that communicate over the Internet.

Aglets Workbench is a very versatile tool for creating secure mobile agent-based applications. However, it does not deal with the important issue of implementing coordination, cooperation and coherence in agent-based applications. Aglets can only engage in directed communication as they use the TCP/IP protocol. With the new features of JKQML, Aglets can be an adequate tool for some mobility if it is needed inside the agent community. Another advantage of Aglets is that it is quite simple to use the API, it goes pretty fast to get something done, and a very nice user-interface is provide to control the agents [37].

4. Concordia

Concordia [41] is a Java-based framework for development and management of network-efficient mobile agent applications for accessing information anytime, anywhere, and on any device (see Fig. 8). Concordia offers a flexible scheme for dynamic invocation of arbitrary method entry points within a common agent application. It provides support for agent persistence and recovery and guarantees the transmission of agents across a network. Concordia was designed to provide fairly complete security coverage from the outset.

Within Concordia, an agent's travel plans are specified by its Itinerary. The Itinerary is a completely separate data structure from the agent itself. Concordia provides two forms of asynchronous distributed events: selected events and group-oriented events. The event selection paradigm enables agents to define the types of events they wish to receive. In contrast, group-oriented events are distributed to a collection of agents (known as an event group) without any selection.

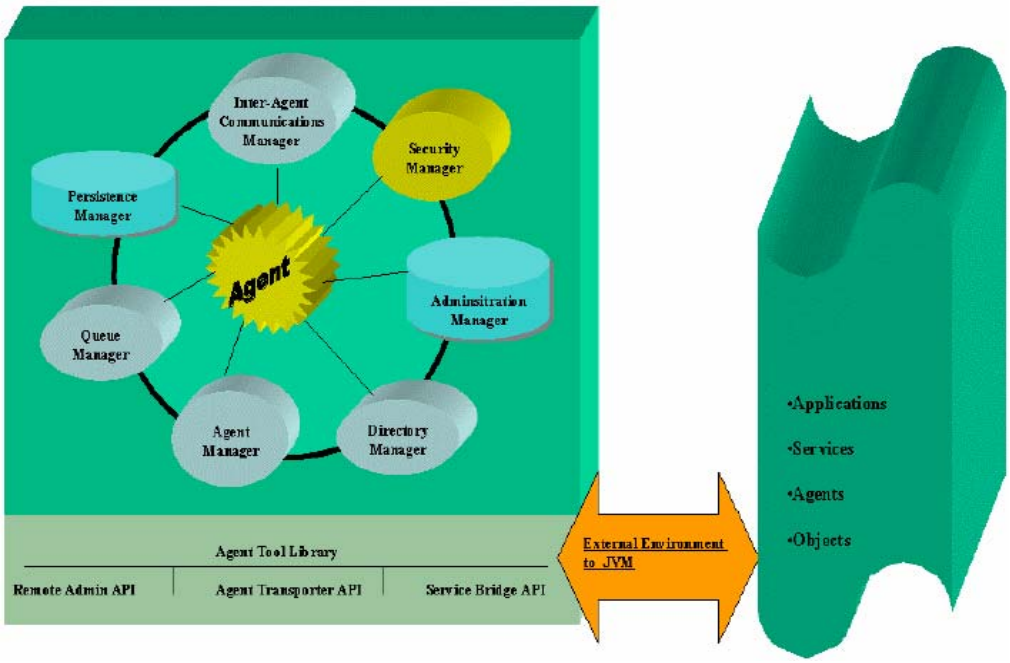

Figure 8. Concordia server architecture [37]

Although Concordia provides a useful set of services for implementing agent mobility, security, persistence and transmission, it does not provide any methodology to

specify how agents in a multi-agent system coordinate cooperate and negotiate to bring about a coherent solution. Emphasis here is on the communication aspect in an agent-based application. Moreover, the fact that the agent Itinerary is outside the agent implies that where the agent travels is maintained in a separate logical location regarding the place where the agent lives. This results in Concordia agents not being totally autonomous [37].

5. Odyssey

Odyssey is an agent system implemented as a set of Java class libraries that provide support for developing distributed mobile applications. Odyssey technology implements the concepts of places and agents. It models a network of computers, however large, as a collection of places. A place offers a service to the mobile agents that enter it. A communicating application is modeled as a collection of agents. Each agent occupies a particular place. However, an agent can move from one place to another, thus occupying different places at different times. Agents are independent in that their procedures are performed concurrently. Odyssey provides Java classes for mobile agents and stationary places.

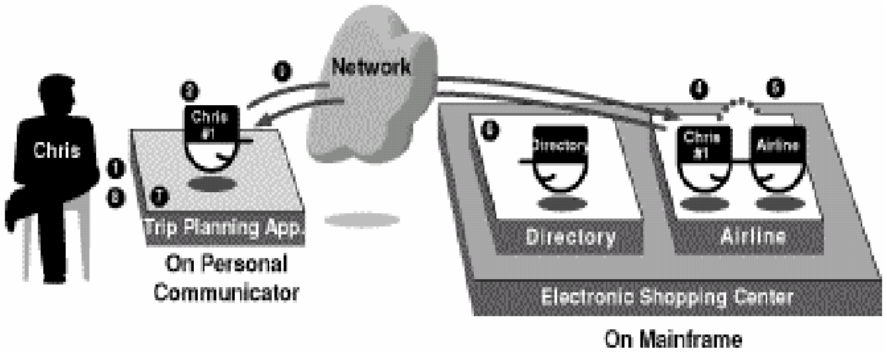

Figure 9. Typical Odyssey application [37]

6. Voyager

Voyager [42] is a Java-based agent-enhanced Object Request Broker (ORB). It allows Java programmers to quickly and easily create sophisticated network applications using both traditional and agent-enhanced distributed programming techniques. It provides for creation of both autonomous mobile agents and objects. Voyager agents

roam a network and continue to execute as they move. Voyager can remotely construct and communicate with any Java class, even third party libraries, without source. It allows seamless support for object mobility. Once created, any serializable object can be moved to a new location, even while the object is receiving messages. Messages sent to the old location are automatically forwarded to the new location.

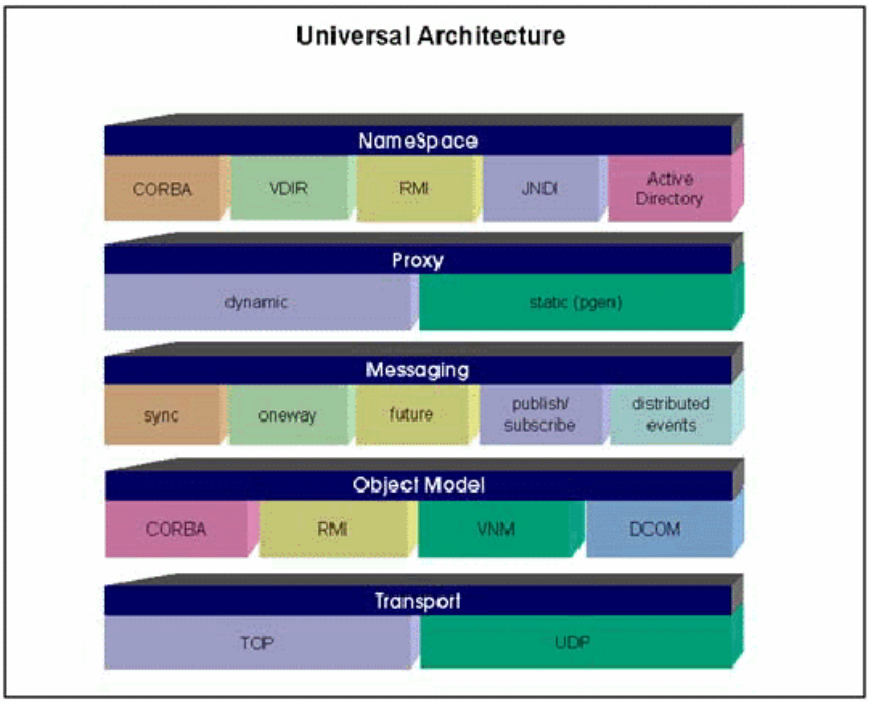

Figure 10. Voyager's universal architecture [42]

Voyager is a very efficient tool for constructing agent-based distributed applications. However, it does not provide any classes for defining the social behavior of agents, does not support broadcast communication and speech-act messaging, and lacks in security.

7. IA Factory

The goal of the IA Factory [43] is to supply the programmer with an Application Program Interface (API) that precludes him from having to go through the entire network programming and debugging. This framework provides a generic agent that allows a programmer to extend its specific behavior. In the simplest case, a table of behavior is sufficient. When an agent requires greater complexity, programmers can extend a class to give an agent complex and interesting behavior.

The IA Factory creates a set of agents, along with an interface to run them. These agents communicate via the KQML language (there is also another version of the library for commercial use that uses XML messages). The agents are lightweight, which means that hundreds of agents can run within one Java Virtual machine. The source code to the agents is generated in Java; hence, they can be customized to increase the functionality of intelligent agents.

8. RETSINA

RETSINA (Reusable Environment for Task Structured Intelligent Network Agents) [44] is an open multi-agent system (MAS) that supports communities of heterogeneous agents. The RETSINA system has been implemented on the premise that agents in a system should form a community of peers that engage in peer to peer interactions. Any coordination structure in the community of agents should emerge from the relations between agents, rather than as a result of the imposed constraints of the infrastructure itself. In accordance with this premise, RETSINA does not employ centralized control within the MAS; rather, it implements distributed infrastructural services that facilitate the interactions between agents, as opposed to managing them.

Following is a graphical representation of the RETSINA MAS:

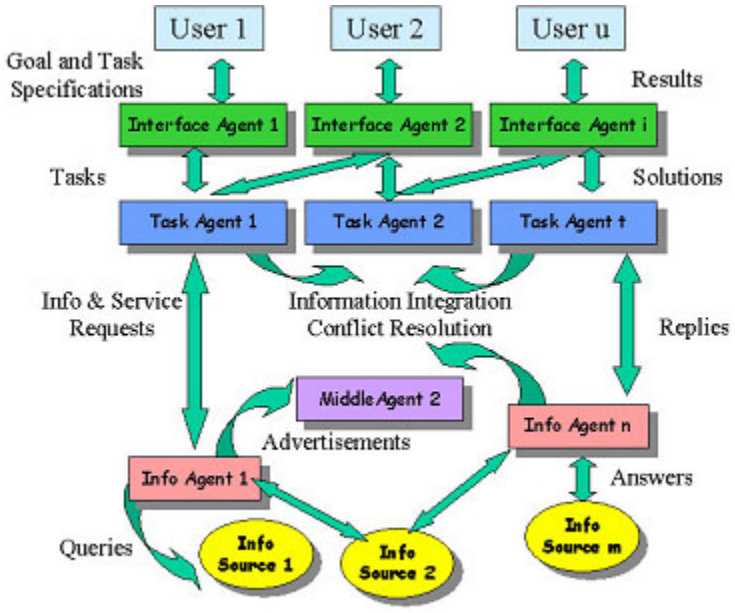

Figure 11. RETSINA MAS [44]

The RETSINA framework is being used to develop distributed collections of intelligent software agents that cooperate asynchronously to perform goal-directed information retrieval and information integration in support of performing a variety of decision-making tasks. A collection of RETSINA agents forms an open society of reusable agents that self-organize and cooperate in response to task requirements. Their designer focused on three crucial characteristics of the overall framework that differentiate RETSINA from others:

- Use of a multi-agent system where the agents operate asynchronously and collaborate with each other and their user(s)
- Agents actively seek out information
- Information gathering is seamlessly integrated with problem solving and decision support

The RETSINA functional architecture consists of four basic agent types:

1. **Interface agents** - interact with users, receive user input, and display results.
2. **Task agents** - help users perform tasks, formulate problem-solving plans and carry out these plans by coordinating and exchanging information with other software agents.
3. **Information agents** - provide intelligent access to a heterogeneous collection of information sources.
4. **Middle agents** - help match agents that request services with agents that provide services.

RETSINA addresses the problem of how to facilitate communication among agents of different types. As part of the RETSINA infrastructure of reusable agents, middle agents represent an important step in our ongoing effort to provide a foundation that will allow heterogeneous agent types and architectures to interoperate successfully. Each RETSINA agent has four reusable modules for communicating, planning, scheduling, and monitoring the execution of tasks and requests from other agents.

a. The **Communication and Coordination** module accepts and interprets messages and requests from other agents.
b. The **Planning** module takes as input a set of goals and produces a plan that satisfies the goals.
c. The **Scheduling** module uses the task structure created by the planning module to order the tasks.

d. The **Execution** module monitors this process and ensures that actions are carried out in accordance with computational and other constraints.

Below is a graphic representation of the RETSINA agent architecture:

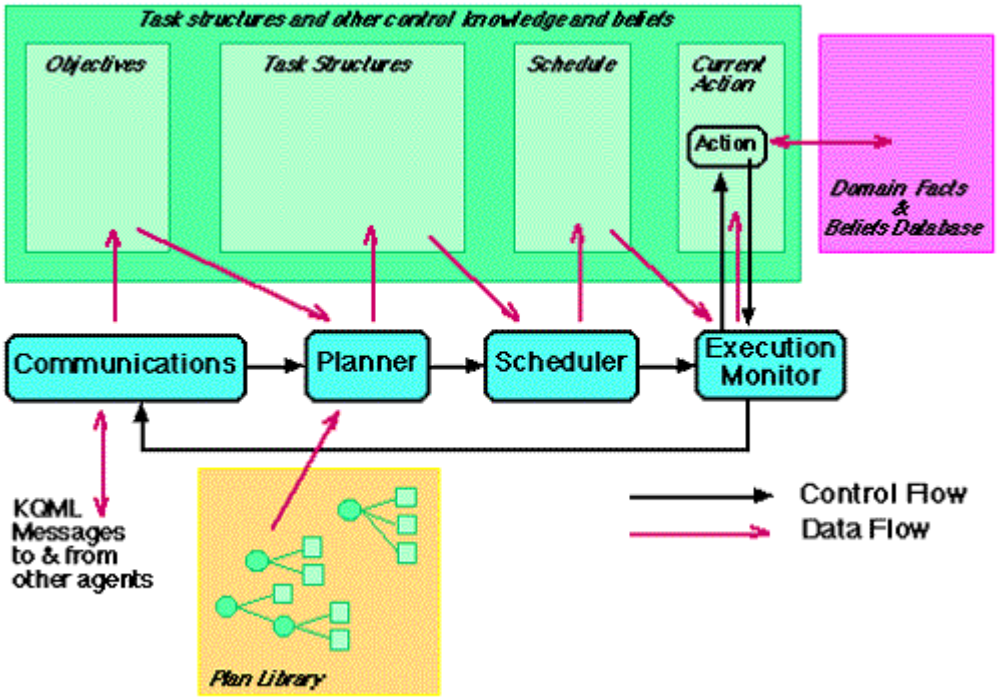

Figure 12. RESTINA agent architecture

The set of Java classes within RESTINA emphasized on how agents can find information or advertise their capabilities. No particular attention is given regarding the behavior of each agent and how they interact with other agents. The Name Server API uses a centralized approach, making the system less scalable and fault-tolerant. As RETSINA is an open system, any agent on the Internet can communicate and interact with the actual community [37].

9. MAST

The MAST (Multi-Agent System Tool) is a heterogeneous, multi-agent, general-purpose MAS. It employs a decentralized model of control and consists of two basic types of entities: agents, defined as autonomous entities that may carry out specific tasks by themselves or in conjunction with other entities, and the network through which they interact. MAST agents consist of a structured set of elements that include services, goals,

resources, internal objects, and control. Control in MAST is merely a specification of how an agent handles a service request. The network consists of a yellow page service incorporated in an agent that facilitates lookup services. MAST is a loosely coupled system of agents that use the Common Knowledge Representation Language (CKRL) to communicate and MAST-ADL (Agent Description Language) to describe agents [45].

The MAST architecture is a very complete multi-agent system tool. However, the fact that it is implemented in C++ makes it less portable and less multifunctional than Java-based frameworks. Many of the services within MAST (interoperability between heterogeneous agents) are unnecessary because they are already included in the Java language [37].

10. dMARS

dMARS is an agent-oriented development and implementation environment designed for building complex, distributed, time-critical systems. It is intended for rapid configuration and ease of integration, and it helps with system design, maintenance, and reengineering. dMARS agents are designed according to the BDI (Beliefs, Desires, and Intentions) model. They are able to reason about their environment, their beliefs, their goals, and their intentions. They model their expertise as a set of context-sensitive plans. These plans can both react to changes in the environment and proactively pursue the agent's objectives. Using dMARS, multi-agent systems can be implemented as lightweight processes within a single UNIX process, as separate UNIX processes on the same machine, or as a distributed configuration communicating over a TCP/IP network. Interfacing with other processes is achieved via a simple, well-defined communication protocol. The system provides comprehensive libraries and components to support the development, implementation and testing of an application, therefore minimizing the need to develop application-specific support code.

dMARS is written in C/C++, thus, does not provide for true architecture neutrality and portability. Unlike Java-based systems, applications developed in dMARS can only run on limited platforms and require using different compilers for different platforms. It supports only a limited number of C++ compilers.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. ENTERPRISE NETWORK MANAGEMENT IN THE DEPARTMENT OF DEFENSE

The purpose of this chapter is to give the reader a top-down overview of the requirement for enterprise network management within the Department of Defense (DoD). The first part provides the background of the vision to transform the military into an integrated enterprise-level joint force that is the most technically superior and capable in the world. The study then works its way down to discuss the requirements for the enabling network enterprise infrastructure and the impact that it will have on management and control. The final part presents the Army's transformation and efforts to establish and control an enterprise-level network infrastructure.

A. BACKGROUND - TRANSFORMATION OF THE ARMED FORCES

In order to have the capability to fight and win against the asymmetric threats employed today, the United States recognized the need to transform the military. Asymmetric threats mean that there is no clear battle line between opposing forces; the enemy is much smaller, decentralized, clandestine, and wage tactics such as terrorism against non-combatants, and guerilla warfare. Therefore, DoD devised a strategy that leverages state-of-the-art technologies to make the military a more effective and efficient integrated joint force. This strategy is embodied in two linked documents known as the Joint Vision 2010 and Joint Vision 2020.

1. Joint Vision 2010/2020

JV 2010/2020 lay down the conceptual framework for how the United States Armed Forces will transform into a futuristic, highly technical, superior joint force. It envisions the development of a superior joint force that is dominant across the full spectrum of military operations – persuasive in peace, decisive in war, preeminent in any form of conflict [1]. The integration of core competencies provided by the individual services and components is essential to establishing a superior joint force. This means that the force must be fully joint – intellectually, operationally, organizationally, doctrinally, and technically.

The Joint Vision is characterized by seven layered concepts that lead up to full spectrum dominance (see Fig. 13). The seven layered concepts are: decision superiority,

information superiority, network-centric warfare (NCW), and the global information grid (GIG). Each of the respective concepts, starting with the GIG, provides a distinctive capability and is the foundation that enables the above layer. These individual layers are the building blocks that collectively enable full spectrum dominance.

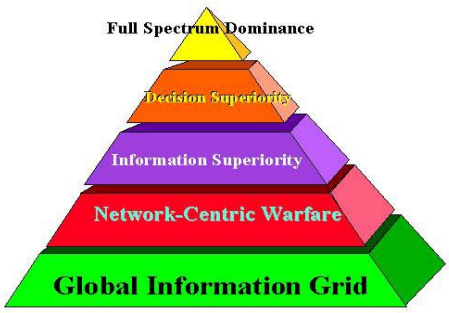

Figure 13. Achieving Full Spectrum Dominance [38]

As shown in Figure 13, each of the underlying layers is a foundation that enables the above layer. Information superiority is the key enabler to achieve decision superiority for the warfighter, which ultimately leads to full spectrum dominance. Further, information superiority is enabled by the NCW concept, which requires the aggregation and interoperability of the stovepiped networks, legacy systems, and applications. The GIG is the underlying infrastructure that integrates the networks and systems that enables NCW.

The foundation of this research study is derived from the envisioned technology requirement necessary to achieve the JV2010/JV2020 concepts. A pivotal aspect of the transformation entails an insurgence of state-of-the-art technology insertions to move from a platform-centric capability to a network-centric capability, and ultimately to a knowledge-centric capability over the next quarter century. A platform-centric capability focuses on individual platforms (weapons systems, information systems, etc.) that are networked and managed within centralized intranets. The network-centric capability is

based on fusing the platform-centric stovepipes into a single, fully integrated, and decentralized network for sharing information across platforms, thus, flattening the command and control hierarchy. The knowledge-centric capability will evolve the network-centric capability to allow generated knowledge to be distributed and shared throughout the military enterprise.

2. Information Superiority

To reach full spectrum dominance the transformation of the joint force depends upon information superiority as the key enabler. Joint Publication 1-02 defines information superiority as the capability to collect, process, and disseminate an uninterrupted flow of information while exploiting or denying an adversary's ability to do the same. The essence of information superiority is further elaborated in JV2020 [1]:

> Information superiority provides the joint force a competitive advantage only when it is effectively translated into superior knowledge and decisions. The joint force must be able to take advantage of superior information converted to superior knowledge to achieve "decision superiority" – better decisions arrived at and implemented faster than an opponent can react, or in a noncombat situation, at a tempo that allows the force to shape the situation or react to changes and accomplish its mission.

In order to gain information superiority within the context of full spectrum dominance, the joint force must be fully synergized in terms of information and intelligence sharing, and situational awareness. The interconnecting of platforms into a single shared awareness environment to achieve information superiority is the underlying principle of network-centric warfare.

3. Network Centric Warfare

According to the pioneers [36], Network Centric Warfare focuses on the combat power that can be generated from the effective linking or networking of the warfighter enterprise. It is characterized by the ability of geographically dispersed forces to create a high level of shared battlespace awareness that can be exploited via synchronization and other network-centric operations to achieve commanders' intent.

Figure 14. Network Centric [3]

The information advantage gained through the use of NCW allows a warfighting force to achieve dramatically improved information positions, in the form of common operational pictures that provide the basis for shared situational awareness and knowledge, and a resulting increase in combat power. The ability to achieve shared situational awareness and knowledge among all elements of a joint force, in conjunction with allied and coalition partners, is increasingly viewed as a cornerstone of transformation to achieve future warfighting capabilities [5].

One of the essential key concepts of the NCW definition is the "effective linking or networking" among entities in the battlespace. This means that dispersed and distributed entities can generate synergy, and that the responsibility and work can be dynamically reallocated to adapt to the situation. The effective linking requires the establishment of a robust, high-performance information infrastructure, or infostructure, which provides all the elements of the warfighting enterprise with access to high quality information services [36].

Establishing a robust, high-performance infostructure is where the realization of NCW gets complicated. Establishing the infostructure implies the integration of heterogeneous, legacy, and proprietary systems, the establishment of a single enterprise network, and the expansion of network control. The military must figure out how to integrate the disparate systems in order to monitor and control it at the enterprise. The gist of this study is that the limitations of the incumbent NM protocols will eventually stifle the migration to full-blown NCW. The traditional protocols will encounter

problems with scalability, reliability, flexibility, and adaptability that are essential to NCW.

The DoD's concept to establish an underlying robust, high-performance infrastructure to effectively link the entities in the battlespace is known as the Global Information Grid (GIG). The next section discusses the GIG in detail.

4. Global Information Grid

The idea of a GIG was originated as a result of growing concerns regarding interoperability and end-to-end integration of automated information systems, streamlined management, and information infrastructure investments within the military. However, the real demand for a GIG was driven by the requirement to achieve full spectrum dominance as expressed JV2010/2020. The GIG provides the enabling foundation for NCW. The success of the GIG will depend in large part on how well it helps achieve force-wide information sharing.

The GIG vision is outlined below [4]:

- A single secure grid providing seamless end-to-end capabilities to all warfighting, national security, and support users
- Supporting DoD and Intelligence Community (IC) requirements from peace time business support through all levels of conflict
- Joint, high capacity netted operations
- Fused with weapons systems
- Supporting strategic, operational, tactical, and base/post/camp/station
- Plug and Play interoperability
    - Guaranteed for US and allied
    - Connectivity for coalition users
- Tactical and functional fusion a reality
- Information/bandwidth on demand
- Defense in depth against all threats

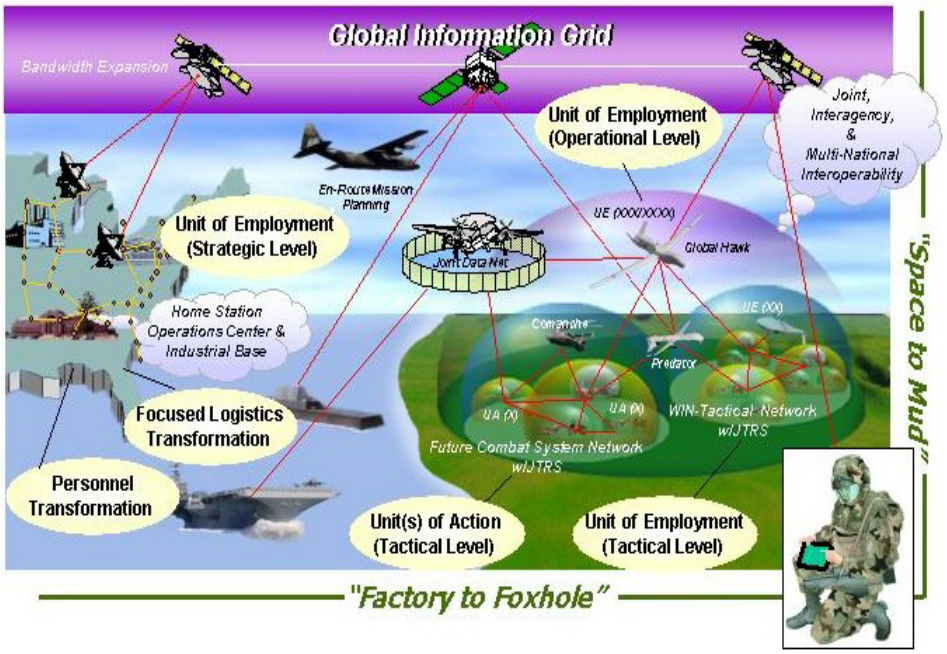

Figure 15. Global Information Grid [47]

On 2 May 2001, the official definition of the GIG was agreed upon and published by the DoD Chief Information Officer (CIO), the Under Secretary of Defense (USD) for Acquisition, Technology and Logistics (AT&L), and the Joint Staff/J6 [4]:

> Globally interconnected, end-to-end set of information capabilities, associated processes, and personnel for collecting, processing, storing, disseminating, and managing information on demand to warfighters, policy makers, and support personnel. The GIG includes all owned and leased communications and computing systems and services, software (including applications), data, security services, and other associated services necessary to achieve information superiority. It also includes National Security Systems (NSS) as defined in section 5142 of the Clinger-Cohen Act of 1996. The GIG supports all DoD, National Security, and related Intelligence Community (IC) missions and functions (strategic, operational, tactical, and business) in war and in peace. The GIG provides capabilities from all operating locations (bases, posts, camps, stations, facilities, mobile platforms, and deployed sites). The GIG provides interfaces to coalition, allied, and non-DoD users and systems.

a. GIG Functions

The explosion of the GIG requirements will increase the demand and criticality of network troubleshooting, network management, dynamic bandwidth management, information and network protection, and spectrum management [6]. Four

defined functions that characterize the information flow and exchange within the GIG are Computing, Communications, Presentation, and Network Operations (NETOPS). Each of these functions imposes a set of complex challenges that must be overcome to realize the full nature of the GIG. However, the scope of this study is mainly focused within the NETOPS functional area. More specifically, this study explores the problems, issues, and solutions for the "network management" aspect of the NETOPS sub-function.

NETOPS is an organizational and procedural framework used to monitor, manage, and control the GIG by means of the sub-functions of Network Management (NM), Information Dissemination Management (IDM), and Information Assurance (IA) [5]. Focusing in on NM, the GIG NM function is defined as is the capability to monitor, control and ensure the visibility of the various networking and internetworking components.

b. GIG Network Management Capabilities Requirements

The Capstone Requirements Document (CRD) [5] for the GIG defines the network management capabilities requirements that are essential to effectively monitor, manage, and control the GIG as an enterprise network. The CRD points out that network management is the set of activities that establishes and maintains the GIG network switching, transmission, information services, and computing resources available to fulfill users' telecommunications and connectivity needs and demands. The CRD further points out that the key GIG NM services are fault, configuration, account, performance, and planning management. The following paragraphs reflect the detailed capabilities requirements for network management as outlined in the CRD. The capabilities requirements are shown in italics within each discussion area.

- GIG End-to-End Situational Awareness. Network managers, on behalf of commanders, must have real-time knowledge of the network. This knowledge must encompass awareness of all aspects of the network, including all network assets, their physical location, and their logical relationship within the network. To accomplish GIG end-to-end situational awareness, systems shall have the NM capability of automatically generating and providing an integrated/correlated presentation of networks and all associated network assets.

- Dynamic, Predictive Planning. Systems shall have the NM capability to perform dynamic, predictive planning by gathering, storing and using knowledge about GIG assets/resources, so as to optimize their utilization. Knowing equipment types and quantities available to support an operation is imperative for GIG utilization planners. Initially, a database must be defined and populated with organizations and their known GIG assets/resources. Once defined and populated, the database should have the capability to be modified, as required, to support changing mission requirements to include activation/deactivation. The network management system should include network design and engineering functions that account for all voice, video, and data networks that could comprise a proposed system, including commercial technology. These functions should include automated mapping of network topology; measurement and recording of traffic flow data; trend analysis; spectrum planning and management; propagation analysis; electromagnetic resolution; and electronic key management. A modeling and simulation capability should be provided to allow a planner to assess the impact of changes to a system or network, without interrupting the operational network. Systems shall have the NM capability to create/modify/distribute GIG network plans and orders in accordance with user requirements.

- Distributed and Partitioned Network Control. Systems shall have the NM capability to transfer control rapidly of one or more objects or groups of varying size, and reestablish control when relinquished without hindering end-to-end visibility by the senior network manager, while maintaining continuous control. Only one designated active manager for a network object should be permitted at any given time. However, oversight of managers of network objects may shift as forces/assets are apportioned, allocated, or assigned without requiring a change of the active manager.

- Remote Object and Network Control and Configuration. Network managers must be able to monitor, configure, and control all aspects of the

network and observe changes in network status. Networks comprising the GIG are evolutionary in nature and generally are comprised of both legacy and emerging systems, some with their own management systems. Systems shall have a NM capability that leverages existing and evolving technologies and has the ability to perform remote network device configuration/reconfiguration of objects that have existing DoD JTA management capabilities.

- Network Status. Components of the GIG provide metrics to network managers to allow them to make decisions on managing the network. Systems shall have an automated NM capability to obtain the status of networks and associated assets in near real time 99% (Threshold, Key Performance Parameter - KPP) and 99.9% (Objective, KPP) of the time.
- Automated Fault Management. Systems shall have the NM capability to perform automated fault management of the network, to include problem detection, fault isolation and diagnosis, problem tracking until corrective actions are completed, and historical archiving. This capability allows network managers automatically to monitor and maintain the situational awareness of the network's manageable devices, and to become aware of network problems as they occur based on the trouble tickets generated automatically by the affected object or network. Alarms will be correlated to eliminate those that are duplicates or false, initiate test, and perform diagnostics to isolate faults to a replaceable component.

The military faces many, many challenges ahead in achieving the aforementioned network management capabilities requirements. There are a number of obstacles, such as scalability issues, that will require innovative solutions in order to accomplish these requirements and bring the GIG to fruition at a cost that is affordable.

c. GIG Network Management Shortcomings

The military is inundated with an overwhelming number of disparate information systems, applications, and network architectures. This dilemma has plagued the military for years due to the lack establishing a single standard. The effort to

integrate these systems into a coherent GIG will by no means be a simple endeavor. An even greater issue is finding or developing a single network management platform that can manage and control these incompatible systems and display a network common operational picture (NETCOP) to maintain situational awareness of the GIG. The CRD identifies a number of critical network management shortcomings that can impede the GIG implementation. These findings are consistent with the shortcomings discussed in Chapter II, and reinforce the need for a more sophisticated enterprise NM solution:

- There is a lack of asset visibility resulting in an inability to effectively manage the overall network to support common user needs. The limited network visibility is significantly impacted by the large number of stovepiped and legacy systems. Stovepiped and legacy systems are normally not designed to support global, end-to-end network management or adhere to a prescribed set of standards for interoperable use across DoD and the Intelligence Community. However, there are dedicated/specialized systems that are required to accomplish specific command missions, but do not support or facilitate effective network management of these systems.
- There are no common prescribed standards for common user systems/networks that would facilitate network management across DoD/IC. This shortfall precludes effective network management, which is essential to ensure the most efficient and effective exchange of information across the battlespace.
- There is no distributed network management capability that would allow the management of common user networks from more than just one central location.
- Existing network management is currently unable to provide a fully integrated multi-level security network.

- DoD has little or no network management capability to accompany its increasingly widespread use and application of advanced mobile wireless computing and networking which are inherently ad hoc.
- There is no prescribed standard joint network management capability for JTF component-level common user systems/networks. Deployed network management suffers from a loosely federated approach for employing government unique (formerly government-off-the-shelf (GOTS)) and COTS software.
- Current end-to-end communications, especially in the last tactical mile, are not fully integrated and interoperable. Specific issues include heterogeneous network design, inconsistent firewall implementations and varying network management policies and tools.

This list of shortcomings clearly identifies the complexities that the military will encounter in the pursuit of the GIG. Attempting to standardize the systems across the enterprise is unrealistic and too costly. Additionally, even with standardization there will still be such issues as scalability. Therefore, the military will have to look beyond the current technologies for viable solutions. The CRD offers no suggested solution, although that is not the intent of the CRD anyhow.

d. GIG Challenge

As noted in the shortcomings above, as well as the numerous complexities presented in Chapter II, realizing the full implementation of the GIG is much easier said than done. Within the purview of this study, the SNMP standard required by the JTA currently lacks the robustness, scalability, reliability, flexibility, and adaptability necessary to meet the requirements as outlined above. However, it is important to understand that the NM challenge of the GIG extends beyond the arguments made in this study. For example, not all desirable managed devices have standard MIBs embedded in them. Devices such as personal computers, personal digital assistants (PDAs), and mobile devices don't usually have MIB structures loaded. Small, lightweight devices such as PDAs are resource restrictive and therefore in many cases manufacturers prefer not to embed them. However, there is nothing to preclude them from being loaded [28].

Bordetsky and Dolk [46] address another important GIG challenge regarding the complexities inherent in the emerging wireless management that is also encumbered within the GIG. Relative to this study, they discuss the issue of managing an increasing number of SNMP MIB objects. They present a scenario regarding the magnitude of the one million customer base wireless only web users of the Sprint PCS system:

> In January 2001, Sprint PCS announced that its customer base for wireless only web users reached the one million mark. With Palm Pilots rapidly merging wireless services, we can picture each emerging mobile user as a node with 2-5 terminal devices. Typically each mobile terminal appearing in the Mobile Switching Center Registries requires on the order of 20 objects, so an approximation of the complexity of a network comparable to Sprint PCS would range from 40 – 100 million objects. And this is only one of the four wireless technologies.

This scenario underscores another difficult challenge for GIG implementation using traditional standards as elaborated in Chapter II. The military must consider the diverse technologies such as the emerging wireless technologies and figure out how to converge them as well as managing the magnitude of management objects. Bordetsky and Dolk offer a Knowledge Management solution found in [46].

V. ENTERPRISE NETWORK MANAGEMENT IN THE ARMY

This chapter provides an overview of the Army's plan to transform itself in accordance with JV2010/2020. The chapter focuses on the Army's enterprise network management approach to give some insight on how the Army plans to operate and implement it. This gives the reader a feel for the inherent complexities and what is required for enterprise-level network management in the Army.

A. BACKGROUND

In line with the full spectrum dominance concept in JV2010/2020, the Army developed a strategy called "Army Knowledge Management" (AKM) to transform itself into a network-centric, knowledge-based, Internet age enterprise force. Conceptually, it will improve information access and sharing, while providing the information infrastructure (infostructure) capabilities across the Army. AKM is intended to improve decision dominance by dramatically enhancing the warfighter's ability to distribute, process, fuse, and correlate unprecedented amounts of actionable data into information – securely, reliably, and quickly enough to enable leaders to synchronize and mass effects for decisive results [22].

The AKM strategy consists of five goals:

1. Leverage all information technology capabilities of the Department of Defense and the Army at the enterprise-level to reduce the total cost of ownership.

2. Integrate best business practices to promote Army transformation to a net-centric, knowledge-based force.

3. Manage the information technology infrastructure as an Army enterprise.

4. Implement a web-based enterprise knowledge portal to provide universal, secure access for the entire Army.

5. Harness the human capital of our Army IT workforce through effective recruitment, training, and retention of our soldiers and civilians.

In accordance with AKM goal 3, the Secretary of the Army published the "AKM implementing guidance Goal 3," dated 18 September 2001, directing the Army to establish the Network Enterprise Technology Command/9th Army Signal Command (NETCOM) to act as the single Army network operator and defender. NETCOM is the Army's single authority to establish, operate, and manage the Army enterprise infostructure (AEI). NETCOM has technical command and control and configuration management authority for the Army's critical networks and systems, and will have operational review/coordination authority for any standards, system, architecture, design, or device that impacts enterprise-level Army infostructure and Network Operations (NETOPS) [24]. Accordingly, NETCOM has assumed technical control of all Army networks – Active, Guard, and Reserve.

B. ARMY ENTERPRISE INFOSTRUCTURE

The Army Enterprise Infostructure (AEI) [25] is the Army's portion of the Global Information Grid (GIG). The AEI is the underlying backbone that will enable network-centric warfare (NCW) capabilities in the Army. Physically, the AEI is the shared computers, ancillary equipment, software, firmware, hardware, services, people, business processes, facilities and related resources used in the acquisition, storage, manipulation, protection, management, movement, control, display, switching, interchange, transmission, or reception of all types of data or information in any format, including audio, video, imagery, voice, or data.

The AEI includes the sustaining base (posts, camps, and stations) with Wide, Metropolitan, Campus, and Local Area Networks (WAN/MAN/CAN/LAN) that extends from the sustaining base to the tactical environment. The AEI is not one contiguous network; it consists of those portions of the GIG that the Army is responsible for operating and managing. It encompasses the Command, Control, Communications and Information Management (C4IM) platforms and services supporting Army users, both in permanent stations and deployed. This includes the Active Army, the Army Reserve and the Army National Guard.

The scope of the AEI encompasses Army information services worldwide access to the GIG. This includes the Top Secret/Sensitive Compartmented Information (TS/SCI) domain and/or networks, Army Secret security domain, Army Sensitive but Unclassified

security domain, Army public information sites on the Internet; and a variety of telephony systems (e.g., Defense Red Switch Network and Defense Switched Network). The AEI includes core services such as email, web, file and print servers, directories, Army Knowledge Online (AKO), Public Key Infrastructure (PKI)/ Common Access Card (CAC), and Army enterprise applications such as personnel and logistics. Figure 16 illustrates the physical scope of the AEI.

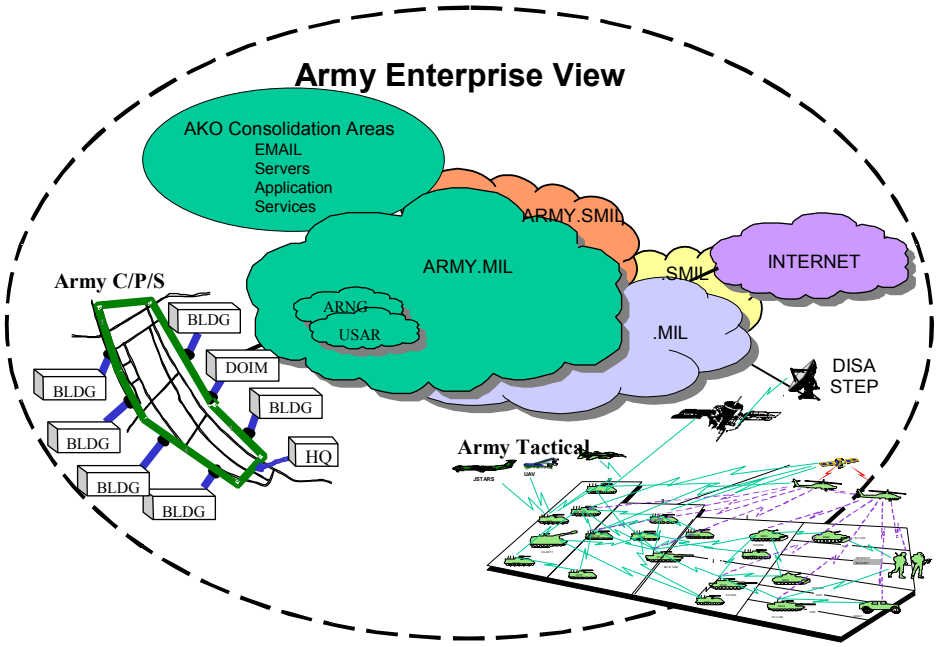

Figure 16. Army Enterprise Infostructure [25]

The current Army sustaining base operating environment consists of a variety of geographically dispersed small, medium, and large installations and user sites (e.g., Reserve Component Centers, Armories, and Recruiting stations), in the Continental United States (CONUS) and outside the Continental United States (OCONUS). Small installations are characterized by having less than 5,000 users, medium installations with 5,000 - 15,000 users and large installations having in excess of 15,000 users. These installations host several IT facilities providing support to not only Army users but also to other Service/Agency tenants. In addition, the Army has many mobile users that require

access to the AEI from outside the enclave. Thus, multiple separate communities of interest with varied IT requirements are found on most Army installations.

Under NETCOM, many disparate operations will be transitioned to consolidated operations and management. There are large headquarters complexes, such as Department of the Army headquarters in the Pentagon. There are several commands that have facilities located around the world in remote locations, such as the Army Corps of Engineers, and Space and Missile Defense Command. Some organizations have activities and isolated users located in other government and commercially leased facilities throughout the CONUS (Continental United States) and in some cases overseas (OCONUS).

Deployed Army units access the GIG through Standard Tactical Entry Points (STEP), commercial satellite, and terrestrial communications links, all managed and operated by the Defense Information Systems Agency (DISA). Deployed units operate and maintain internal networks that provide data, voice, imagery, and video support. These internal networks are referred to as the Tactical Internet (TI) and currently consist of Tri-Services Tactical Equipment, Mobile Subscriber Equipment, Tactical Satellite, and the Enhanced Position Location Reporting System. In the future, the TI will be replaced by the Warfighter Information Network –Tactical and the Joint Tactical Radio System.

C. NETOPS

The NETOPS [25] concept is an organizational, procedural, and technological construct for ensuring information superiority and enabling speed of command for the warfighter. It is defined as the operation and management of the AEI – the organizations, procedures, and technologies required to monitor manage, coordinate, and control the AEI as the Army portion of the GIG. NETOPS links together widely dispersed network operations centers (NOCs) through a command and organizational relationship. It establishes joint tactics, techniques, and procedures to ensure a joint procedural construct, and establishes a technical framework in order to create a Network Common Operational Picture (NETCOP).

1. NETOPS Goals

Below are the established Army goals for NETOPS:

- Enable universal (and secure) access to authorized infostructure services to all Army customers within the Army infostructure - secure single sign-on "plug & play" capability
- Accurately display a total and integrated Situation Awareness of the AEI
- Predict impacts on the AEI of new/changed systems and operational contingencies
- Redirect and reallocate AEI resources in near real-time to support Army response to crisis or unplanned event anywhere within the Army infostructure Operational Area (AOR)
- Provide a consistent, robust, base-level of infostructure services to all authorized Army customers at the least cost feasible within Army operational constraints
- Provide additional (above base level) infostructure services to Army customers on a reimbursable basis
- Perform continuing and non-intrusive technology insertion to improve service levels or reduce cost of providing current base-level services
- Provide Continuity of Operations Plan capabilities

In order to achieve these goals, the Army has set out to standardize and consolidate all the network operations across the infrastructure at the enterprise. By taking this approach the Army intends to increase the quality of service provided to the end-users, better utilize personnel required to perform operational tasks, and minimize the total cost of providing infostructure services. However, consolidation of routine functions causes the physical execution of those functions to move away from the physical proximity of the supported users. Therefore, the Army intends to establish comprehensive remote management capabilities by maintaining consolidated support areas at the installation level. The goal here is to maintain a high level of IT support and service at the installation but to reduce the inherent redundancies and inefficiencies of the current structure. In the future, the consolidated support will gradually shift to a level above installation and then ultimately to the enterprise-level.

2. Systems & Network Management

The NETOPS concept is composed of three integrated mission areas:

- Systems & Network Management (S&NM)
- Information Dissemination Management (IDM)
- Information Assurance (IA)

Together these mission areas facilitate the implementation of "Service Assurance." This approach provides "Assured Network Availability", "Assured Information Protection", and "Assured Information Delivery" at the strategic, operational, and tactical levels through a co-evolution of doctrine, processes, and technology [25]. Figure 17 depicts the NETOPS mission areas and functions.

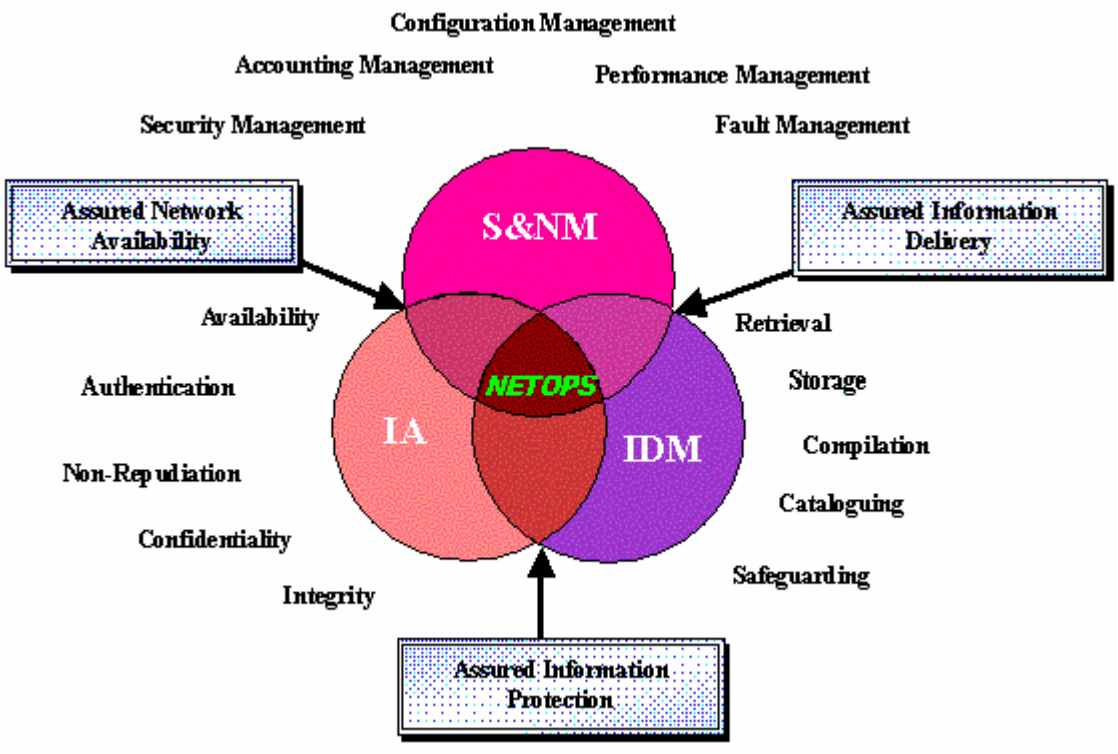

Figure 17. NETOPS Mission Areas and Functions [25]

Systems & Network Management is the management of the network and the devices connected to the network. It is the sum of three management areas: Network Management (to include devices, servers, storage devices, and end-user devices like

printers, workstations, laptops, and handheld computers), Satellite Communications (SATCOM) Management, and Frequency Spectrum Management.

The NETOPS concept was designed to account for the basic network management (NM) functions of fault, configuration, accounting, performance, and security management (FCAPS). This includes systems and applications management and comprises all the measures necessary to ensure the effective and efficient operations of networked systems. Network Management is the only area among the NETOPS function management areas that is germane to this study, hence, it is the only area covered.

3. Army Network Operations and Security Center

The Army Network Operations and Security Center (ANOSC) is NETCOM's central agency responsible for executing enterprise network operations and defense. The ANOSC provides worldwide operational and technical support for the AEI. It is responsible for reporting AEI situational awareness to the Army command and DoD NETOPS. Additionally, on behalf of NETCOM, ANOSC interfaces with all internal and external NOSC for AEI coordination.

The ANOSC oversees subordinate NOSCs that provide distributed networks operations and security within specific geographical area of responsibilities (AOR). Currently, each Army theater AOR has a Theater NOSC (TNOSC), including one in CONUS (C-TNOSC). However, the intent is to move all NOSC and NOSC-like facilities under NETCOM operations and management as part of NETCOM's single authority for operations and management function. The TNOSC acts as a single point of contact for Army network services, operational status, and anomalies in their respective theater AOR. It is also responsible for providing visibility and status information to the ANOSC and DISA Regional NOSCs. Network operations and security responsibilities are further decentralized within the TNOSCs (see Fig. 18).

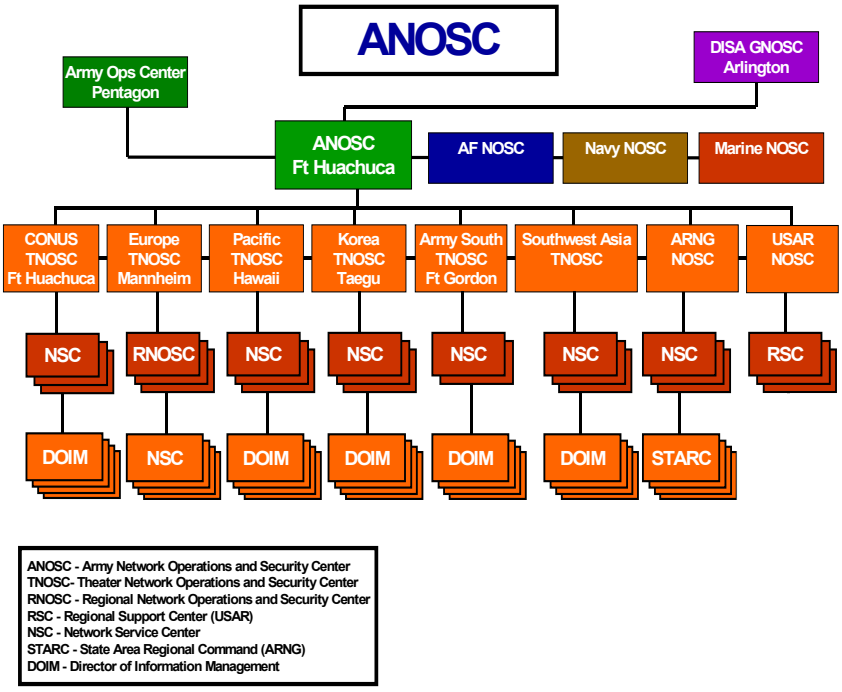

Figure 18. ANOSC and NOSC Relationships [25]

D. NETWORK COMMON OPERATIONAL PICTURE

A key aspect of enterprise network management is the ability for the Army to have comprehensive situational awareness of the AEI. As part of the GIG initiative, the Deputy Secretary of Defense directed all of the services to create and maintain a Network Common Operational Picture (NETCOP) [25].

The NETCOP is an integrated capability that receives, correlates, and displays a view of voice, video, and data telecommunications networks, systems, and applications at the installation/tactical, region, theater, and global levels through the installations/deployed tactical forces, Network Service Centers (NSCs), TNOSCs, and ANOSC respectively. At each level the NETCOP reflects status, performance, and information assurance. At a minimum, the NETCOP requires: telecommunications, system, and application fault and performance status; and significant information assurance reports such as network intrusions or attacks.

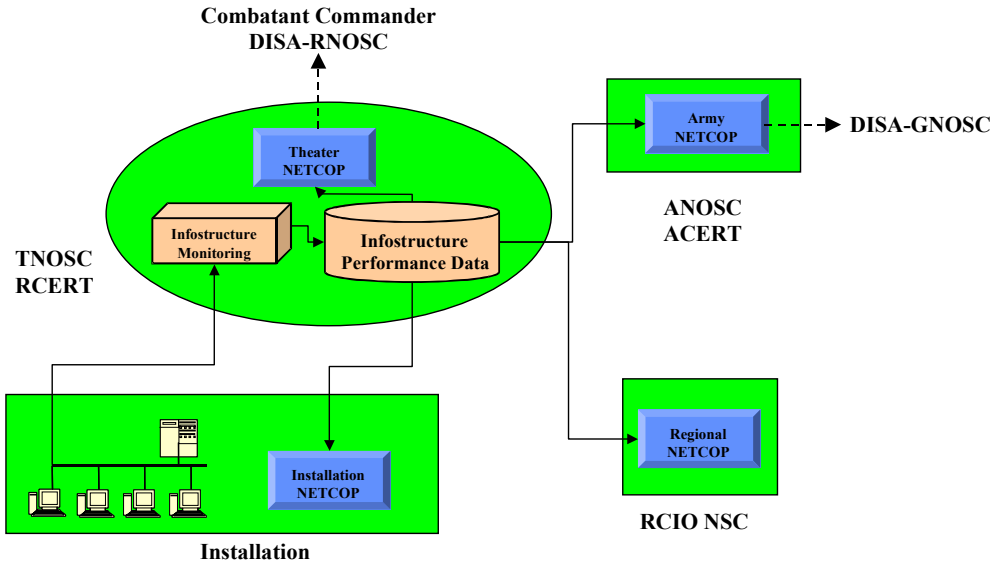

Figure 19. Dissemination of NETCOP Information [25]

The NETCOP provides the ability for Combatant Commanders, Service Components, Sub-unified Commands, Joint Task Forces (JTFs), and deployed forces to rapidly identify outages and degradations, network attacks, mission impacts, C4 (Command, Control, Communications and Computers) shortfalls, operational requirements, and problem resolutions at the strategic, operational, and tactical levels. Figure 19 illustrates the conceptual process by which the NETCOP will be distributed to the various organizations that have a need for this information.

E. SUMMARY

This chapter provides a high-level overview of how the Army is attempting to establish an enterprise network management operation to manage and control the AEI. Hopefully the reader gained a sense of the magnitude of this monumental endeavor. There are tens of thousands of devices that the NETCOM will have to control as an enterprise. A recent C-TNOSC briefing entitled "CONUS NETOPS Status and Issues" [48] identifies some of the challenges. NETCOM only has limited control of network devices: "direct monitoring of devices is limited to those devices directly managed by the

C-TNOSC…" Additionally, the Army NETCOP, which has been online for a year, relies on network management information, such as network outages, to be pushed from the bottom up instead of being controlled at the enterprise. This is tied to the limited number of devices that are directly managed. The presentation also points out the huge level of effort involved in managing at the enterprise level (see Figure 20).

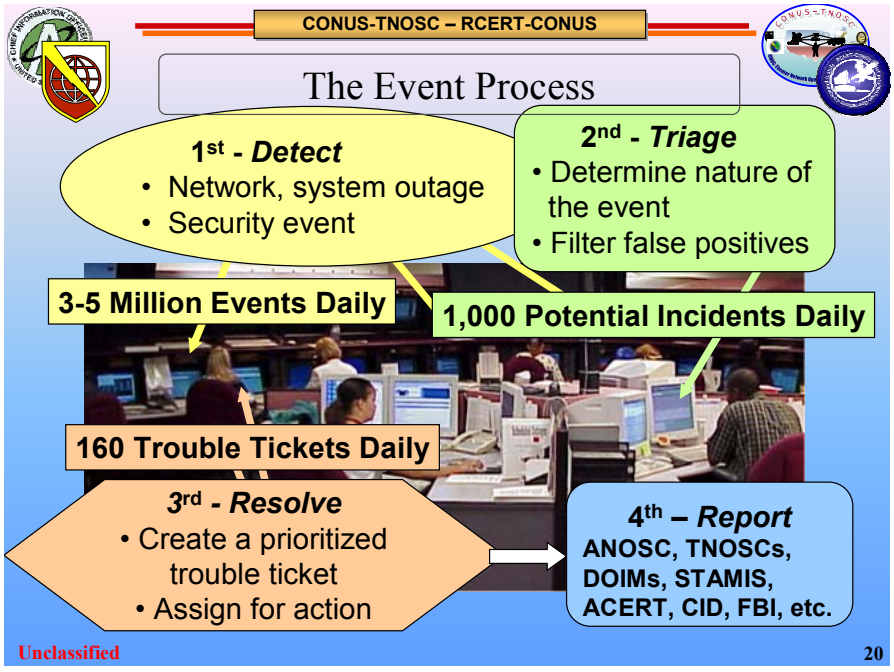

Figure 20. A day in the CTNOSC [48]

Figure 20 depicts a typical day monitoring the AEI in the C-TNOSC. The C-TNOSC is inundated with up to 5 million events daily that have to be filtered. Once filtered, trouble tickets must be assigned for corrective action and reported. This is a highly intensive and costly operation that requires a number of people, equipment, processing power, and bandwidth.

The labor-intensive operation as described in the previous paragraph exemplifies the insufficiency of the traditional protocols such as SNMP. For instance, SNMP simply gathers the data and reports it for handling by human operators. While it does off-load some of the computation load, it has no capability to off-load the four step event process described above. The idea of intelligent-agent-based network management is to

significantly reduce the intensity and cost by allowing intelligent agents to carryout many of these functions in a more effective and efficient manner.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. CONCEPTUAL APPROACH

This chapter presents a conceptual intelligent-agent-based architecture to emphasize the potential of intelligent-agent-based technology for Army enterprise network management. The first part gives an overview of the Control of Agent-Based Systems (CoABS) multi-agent system that serves at the infrastructure for the conceptual architecture. The second part of the chapter describes the details of the conceptual architecture.

A. CONTROL OF AGENT-BASED SYSTEMS

Control of Agent-Based Systems (CoABS) [51] is a research program of the US Defense Advanced Research Projects Agency and the US Air Force Rome Labs. The program is aimed at developing and demonstrating techniques to safely control, coordinate and manage large systems of autonomous software agents. The program investigates the use of agent technology to improve military command, control, communication, and intelligence gathering by enhancing the dynamic connection and operation of military planning, command, execution, and combat support systems to quickly respond to a changing operational picture.

Over twenty universities and companies are participating in the CoABS research effort. Each participating organization brings to the program its own agent architecture, with different agent communication languages, ontologies, and agent-based services. Some of the organizations and architectures involved include RESTINA agents from Carnegie Mellon Univesity, TEAMCORE agents from the University of Southern California Information Science Institute, D'Agents from Dartmouth College, EMAA from Lockheed Martin, and Nomads from the University of West Florida. CoABS is a six year DARPA program that will end December 2003.

1. CoABS Background

The CoABS program was initiated to address many of the challenges within today's military environment. The CoABS perspective of the military environment is viewed as very dynamic, with operations changing quickly, hardware and software moving, connecting, and disconnecting, and network bandwidth availability varying greatly. There are essential, but inflexible, stove-piped legacy systems that need to be

integrated. There is information overload, with vastly increased data available, but inadequate tools to filter the data. Lastly, the military environment is heterogeneous with multiple standards and interfaces, as well as multiple hardware and software platforms [49].

To resolve the inherent complexities of within the military environment, the CoABS visionaries realized that the military needed a customized software technology solution that could be rapidly developed at a low cost and execute on readily available hardware without overloading conventional processors. This need is what triggered the CoABS concept - leverage previous research in distributed artificial intelligence to develop intelligent agents that will handle the complexities more efficiently and effectively.

The CoABS program also identified the need for cooperation among the heterogeneous agents produced by different developers. Cooperation among agents is critical to building powerful applications to support military capability. Without cooperation, monolithic agents would have to handle each new task. Control strategies are needed to build small teams of agents that can cooperate in a robust and flexible manner, as well as a very large number of agents that exhibit macro scale behavior without attending to the detailed behavior of individual agents. Furthermore, there are no sufficient algorithms, policies, or mechanisms that prevent a large heterogeneous set of agents from exhibiting dangerous or chaotic behavior on a network. This lack of control could lead to clogged networks, wasted resources, poor performance, system shutdowns, and security vulnerabilities.

The CoABS program setout to develop technologies for the control of multi-agent systems with predictable behavior for automating military command and control in a cost-effective manner. If successful, the systems of cooperating agents and agent ensembles are expected to dramatically reduce the information systems workload for the entire spectrum of military forces from the national command authority down to the small-unit level as well as provide a framework for resource management in a dynamic hostile or unpredictable environment in which software systems are adaptable, self-configuring, self-healing and evolvable.

Specifically, CoABS proposes to develop:

- A simple agent programming methodology supported by sophisticated component libraries that can automate complex functions cheaply and easily with agents assembled from powerful pieces.
- Compatible agent behavior models.
- Interoperable agent communication languages.
- Advanced, fully protective agent services for protecting both agents and hosts/current servers/existing data sources.
- Simple methods of understanding agent behavior.

2. Agent Grids

A grid is fundamentally a mechanism/infrastructure that helps integrate resources. For example a power grid or transportation grid both supply an enabling capability (electrical power, transportation of goods and services) that fulfill the infrastructure needs of diverse businesses. In a computer-related grid, such resources exist at multiple technical levels, and hence grids can exist at these same levels. Brian Kettler [50] explains the various types of grids:

1. **Computational grids** integrate distributed computing resources to address supercomputing, collaborative computing, and on-demand computing applications.
2. **Data grids** integrate diverse types of data into unified, interrelated collections that support complex applications. Current databases illustrate some of the data integration facilities, and the Web illustrates some of the diversity and scope, required in such grids.
3. **Object grids** extend data grids with associated software components, and hence provide unified access to both data and the resources necessary to operate on that data. Distributed object systems and the Web illustrate some of the aspects required in such object grids.
4. **Agent grids** can be thought of as an extension of object grids with "smarter" software. The CoABS grid, explained later, is example of an agent grid.

Kettler further explains that in addition to the need for the individual technical grid levels described above, these individual grid levels need to be unified. An agent-

level grid supporting this requirement should provide both grid capabilities at the computation and data/object levels *in support of* agents, as well as grid capabilities at these other levels *enabled by* agents. Both these types of support are important in making the maximum use of agent-level capabilities. For example, agent-level grids can take advantage of the capabilities of underlying computational grids in supporting their load balancing and quality-of-service requirements (particularly where the higher-level grids can interact directly with the lower levels to exert control). Operational agent grids will also need to interact with data and object systems because much information and software functionality that will need to be accessible to agent grids will continue to exist in these systems. The CoABS Grid is an agent-level grid that encompassed these properties.

3. The CoABS Grid

The CoABS Grid [49] is a framework for federating heterogeneous agent systems designed to meet the challenges of the military environment, as well as address the heterogeneity among the participating agent research communities. The CoABS Grid is an adaptive and robust collection of infrastructure, services, agents, standards, and protocols. It enables the run-time integration and dynamic interoperability of distributed agents, objects, devices, and legacy systems. "You can also think of the Grid as this infrastructure layer and all the agents and services running on it."[51] Although the CoABS Grid is being developed for military application, it is a general-purpose agent framework with potential use by a wide variety of applications, such as enterprise network management.

The CoABS Grid supports dynamic registration and discovery of relevant participants and flexible run-time communications. It includes a method-based application programming interface to register agents, advertise their capabilities, discover agents based on their capabilities, and send messages between agents. Agents can be added and upgraded without reconfiguring the network. Failed or unavailable agents are automatically purged from the registry.

The CoABS Grid has some unique characteristics that distinguish it from other agent infrastructures. The CoABS Grid is transport neutral in terms of agent communication. The CoABS Grid defines a well-known message -delivery interface.

Grid agents must have a proxy that supports the message-delivery interface, but the agent is free to use any transport to communicate with that proxy. In order to send a message to an agent, the sender makes a local method call to an agent proxy. The proxy transfers the message to the actual agent using a protocol private to the agent and proxy. If the transport mechanism is changed, the sender code is not affected. The CoABS Grid provides a means of obtaining agent proxies based on agent characteristics and types.

CoABS Grid communication is fully distributed, in that each agent sending a message communicates directly with the receiver, using the proxy registered by the receiver. That is, all agent-to-agent communication is point -to-point. Thus, as the number of agents on the CoABS Grid increases, agent communication performance is only affected by the distribution of the agents in the network, the network bandwidth, and the details of the particular proxy implementation.

4. Elements of the Grid

At an abstract level, the CoABS Grid consists of four main elements: Applications, Grid-aware components, Grid services, and the Grid infrastructure. Figure 21 depicts the relationship between these elements. *Applications* are shown as red ovals (.e.g., the Master Battle Planner). Most of these are legacy applications but some could be comprised of agents. Each application has a proxy agent that uses what is known as a GridServiceHelper (Grid services are green rectangle) module to interoperate with other components on the Grid and Grid core services, shown in the center of the diagram. Yellow ovals represent various *data sources* that are accessed by the applications to which they are linked. Data is obtained from a variety of collection mechanisms, shown with military equipment icons. Components that can be connected to the Grid (typically via a GridServiceHelper or GridAgentHelper) are called *Grid-aware components* (GAC).

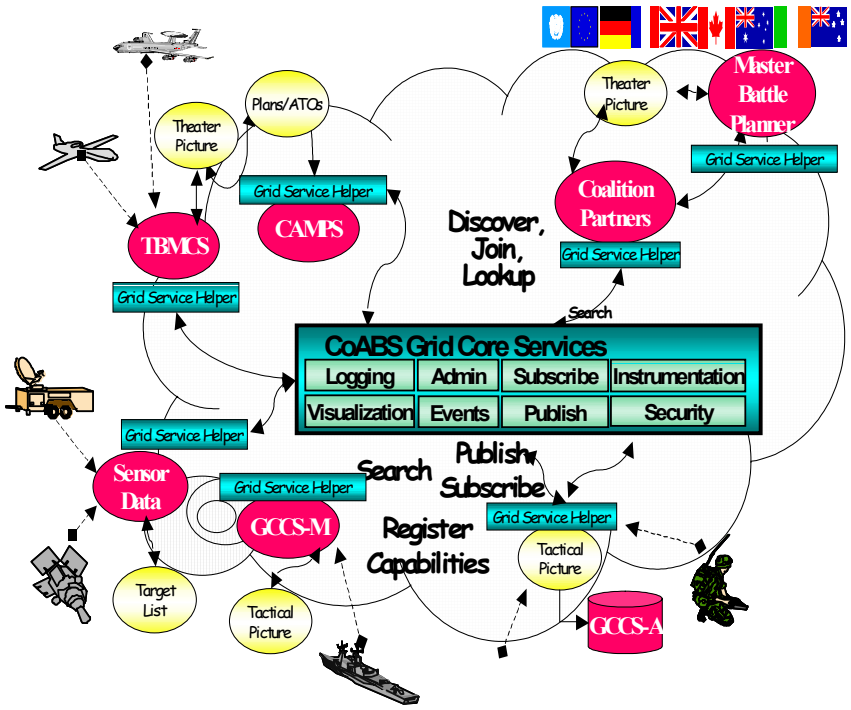

Figure 21. Sample Grid-enabled Application for Coalition Military Operations [50].

Grid services (green boxes in Fig. 21) provide functionality that enables Grid-aware components to interoperate and form applications (e.g., component discovery, brokering, translation, etc.). They also help ensure the smooth operation of applications by providing facilities for instrumenting, testing, debugging, visualizing, and managing applications. Grid-aware components can access Grid services by access mechanisms such as protocols, wrappers, et cetera.

The *Grid infrastructure* is comprised of "lower-level" infrastructures (possibly layered themselves) and mechanisms (protocols, wrappers, etc) that are used by (1) Grid services to talk to one another and (2) by components directly to talk to other components and to access Grid services. The Grid Infrastructure leverages other infrastructures that provide specific connectivity (and services) among components of the same kind. However, the Grid Infrastructure does not necessarily replace the other infrastructures because these architectures are often optimized for the kinds of components they integrate. At the lowest level, there is the network infrastructure (network software, protocols, and network hardware) such as the Internet – or possibly a private intranet. This provides the low-level transport mechanisms between distributed machines [50].

5. CoABS Grid Implementation [50]

This section provides an overview of how the CoABS grid is implemented, describing Grid operations and component interactions. The details within this section were extracted from various the CoABS technical papers found at the CoABS website [49] [50] [51]. Figure 22 below illustrates that CoABS Grid architecture that is described in this section.

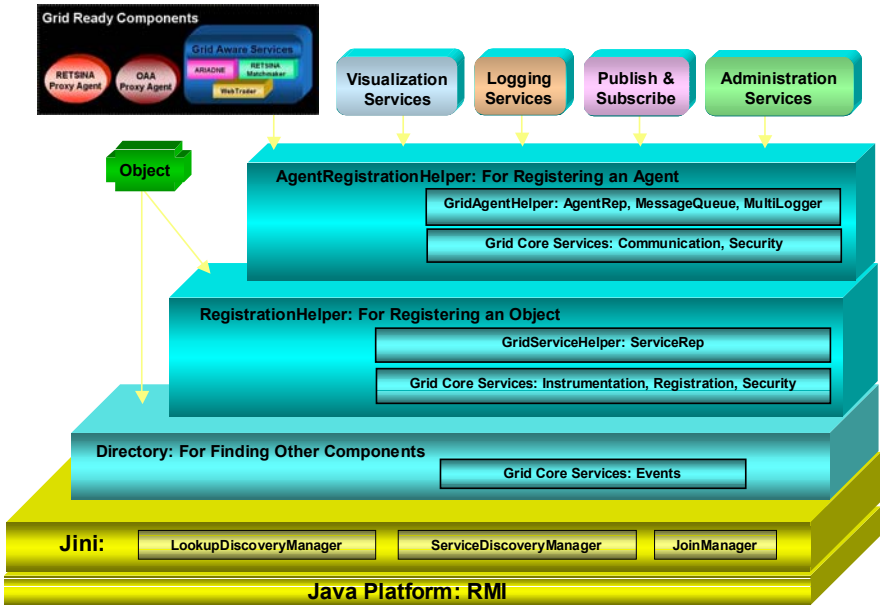

Figure 22. CoABS Grid Architecture [50].

The CoABS Grid is built using the Jini™ Connection technology developed by Sun Microsystems. The robust and dynamic nature of the CoABS Grid is derived from Jini™. The CoABS Grid software is written in Java and also uses Java Remote Method Invocation (RMI) for inter-agent communication. Members of the CoABS research community have created proxies that integrate the CoABS Grid with agent systems written in C++ and Lisp and for the Palm Pilot KVM. The CoABS Grid takes advantage of three important components of Jini™:

1. the Jini™ concept of a service, which is used to represent an agent,
2. the Jini™ Lookup Service (LUS), which is used to register and discover agents and other services, and
3. Jini™ Entries, which are used to advertise an agent's capabilities.

A Jini™ service is a Java object that is serialized and stored in the LUS. The LUS supports lookup of services based on type, attribute values, and unique identifier. When a Jini™ client performs a lookup through the LUS, the service object is returned to the client. The service may optionally be a proxy that uses a remote connection to communicate back to the true service at a different location. The remote connection is transparent to the client and can be of any type, e.g. RMI, CORBA, or secure socket.

The LUS grants leases to registered services, assigns globally unique identifiers to services, and supports lookup of services. It is the service's responsibility to maintain its lease with the LUS, however Jini™ provides helper classes to do this automatically. If a service cannot maintain its lease because of either failure of the service or failure of the network connection between the service and the LUS, the service will be purged from the LUS, so that the LUS contents remain current.

Jini™ provides helper classes that use a multicast protocol to find any LUSs that are running within a local area network. No prior knowledge of the machine name or port that the LUS is running on is required. Jini™ provides a unicast protocol to find LUSs outside the local area network. Service registration is maintained in all local and distant LUSs. The registration is automatically propagated to any new LUS processes that are started. Multiple LUSs can be run for robustness and scalability. If one goes down, the others will still maintain registration and lookup.

Jini™ services are described in the form of a Jini™ Entry. An Entry is a collection of service attributes that is stored in the LUS along with the service. Many Entries can be stored for a single service. Entry templates are used in Jini™ and CoABS Grid lookup methods to match registered services. Entry templates and service types can be used to filter the number of services that are downloaded from a LUS over the network.

The CoABS Grid uses Jini™ Entries for agent capability advertisements. The CoABSAgentDescription Entry has fields for agent name, description, organization, architecture, ontologies, content languages, display icon URL, documentation URL, and unique ID.

The CoABS Grid provides helper utility classes that are local to an agent and that hide the complexity of Jini™. These classes automatically find any LUS in both the local area network and user-designated distant machines. The CoABS Grid supports agent and service discovery based on Jini™ Entries and arbitrary predicates as well as by service type. The CoABS Grid also provides event notification when agents register, deregister, or change their advertised attributes.

The CoABS Grid defines a Jini™ service interface called the AgentRep, which is a proxy to the agent. This interface defines a method called addMessage(), which uses a remote connection to deliver a message back to the agent. Thus, when a client agent calls a CoABS Grid lookup method, a proxy that allows immediate direct communication back to the agent is returned. The client agent can include its own AgentRep in the message it delivers, so that two-way communication can be established with no further lookup. The CoABS Grid is transport neutral in terms of agent communication. The CoABS Grid defines the interface, but the agent proxy is free to use any transport in its implementation.

The CoABS Grid currently provides an AgentRep implementation that uses RMI for message transport. Other transport mechanisms are in development. An AgentRep downloaded to a client is connected to a MessageQueue object local to the agent using RMI. A MessageListener interface is also defined to allow agents automatic notification of incoming messages. Several classes of CoABS Grid messages are provided. Some include text messages only, while others allow data attachments. The CoABS Grid is language neutral; any agent communication language can be used. It is up to the communicating agents to decipher the contents of a message. The CoABS Grid also provides methods to send a message to a group of agents matching a particular template or satisfying a particular predicate.

Agent communication is fully distributed, in that each agent sending a message communicates directly with the receiver, using the proxy registered by the receiver. The sender is unaware of the transport mechanism being used, although currently RMI is the default. Thus, as the number of agents on the CoABS Grid increases, agent communication performance is only affected by the distribution of the agents in the

network, the network bandwidth, and the details of the particular AgentRep implementation. For this reason, this portion of the architecture is thought to be highly scalable.

6. CoABS Experimentation and Findings

CoABS has undergone and is undergoing a number of experiments to test the capabilities of intelligent agents in a multi-agent environment. There are four ongoing Technical Integration Experiments (TIEs) [51] within the CoABS program. Each TIE involves a number of researchers who have come together to solve a particular problem by combining their research efforts. The Coalition Agents Experiment (CoAX) is addressing the unique aspects of achieving coherent Coalition operations from diverse "come-as-you-are" elements. The Mixed-Initiative Agent Team Administration (MIATA) TIE is exploring approaches to human/agent coordination in large, continuous C2 organizations. The Electric Elves TIE is investigating the use of teams of software agents to aid humans in facilitating an organization's coherent functioning and rapid response to crises, while reducing the burden on humans. The Mobility TIE is conducting experiments to determine under what conditions mobile agents should be deployed, as well as building CoABS Grid components to facilitate the movement of mobile agents between heterogeneous mobile agent platforms.

One the more important experiments for a multi-agent system, as well as for intelligent-agent-based network management, is scalability of the infrastructure. In August 2000, the CoABS program conducted an experiment [49] on the scalability of the CoABS Grid Infrastructure with respect to agent lookup. Basically, the experiment investigated how lookup time's scaled as the lookup service became more populated. The experiments were designed to give a qualitative understanding of whether performance problems become apparent with a highly populated LUS.

In the experiments, 500 agents were registered at a time with the CoABS Grid, until a total of 10,000 agents were registered. Twenty-two different agent queries were performed after each group of 500 agents were registered, measuring how long it took to fetch the agents and the number of agents retrieved. The experiments found that sequential lookup scales well to 10,000 agents. Lookups that retrieved a single agent, as well as lookups that matched no agents were not affected by the number of agents

registered. Additionally, time to lookup multiple agents increased proportionally to the number of agents retrieved. Finally, time to lookup multiple agents is independent of the number of agents registered. Experiments performed by an independent subcontractor also found constant lookup times for lookups that returned 120 agents with a maximum of 600 agents registered, and lookups that returned 250 agents with a maximum of 2500 agents registered. Table 1 provides a summary of the experiment results.

Table 1 Lookup Experiment Results Summary [49]

Processor	# Agents (On This Machine)	# Agents (Cumulative)	Least Time to Look Up 1 Agent* (ms.)	Least Time to Look Up 0 Agents* (ms.)	Time to Look Up Blue Agents** (ms.)	Number Blue Agents Found**	Time to Look Up Blue/French Agents*** (ms.)	Number Blue/French Agents Found***
PIII 733 MHz	Jini™ LUS	N/A	N/A	N/A	N/A	N/A	N/A	N/A
PIII 600 MHz	Lookup Tests	N/A	N/A	N/A	N/A	N/A	N/A	N/A
PIII 400 MHz	500	500	90	10	771	39	200	6
PII 500 MHz	500	1,000	50	10	1543	77	310	11
PIII 400 MHz	500	1,500	80	10	1863	116	471	17
PIII 733 MHz	500	2,000	70	10	2664	154	511	22
PIII 733 MHz	500	2,500	80	10	2774	193	661	28
PIII 500 MHz	500	3,000	70	10	3104	231	801	33
PII 300 MHz	500	3,500	70	10	4446	270	831	39
PIII 866 MHz	500	4,000	70	10	4417	308	1002	44
PII 450 MHz	500	4,500	80	10	4947	347	1171	50
PIII 450 MHz	500	5,000	70	10	4917	385	1082	55
PII 400 MHz	500	5,500	70	10	5828	422(424)	1241	61
PII 450 MHz	500	6,000	70	10	5848	462	1302	66
PII 233 MHz	500	6,500	70	10	7361	498(501)	1502	72
PIII 733 MHz	500	7,000	70	10	7191	539	1402	77
PIII 650 MHz	500	7,500	70	10	8142	577(578)	1662	83
PIII 550 MHz	500	8,000	70	10	7681	613(616)	1602	87(88)
PIII 400 MHz	500	8,500	70	10	9393	653(655)	1943	93(94)
PIII 400 MHz	500	9,000	70	10	11377	692(693)	5047	98(99)
PIII 677 MHz	500	9,500	70	10	9604	730(732)	2113	104(105)
PIII 733 MHz	500	10,000	70	N/A	9884	769(770)	2123	109(110)

* Twenty separate lookup tests: looked up agents with size = 0, 500, 1000, 1500, ….. 10,000 – zero or one agent of each size

** One attribute lookup test: find all agents with TestEntry.color = Color.blue

*** Two attribute lookup test: find all agents with TestEntry.color = Color.blue and TestEntry.language = "French"

Figure 23 shows that lookup performance of the LUS appeared to degrade when looking up multiple agents as the LUS was populated with up to 10,000 agents.

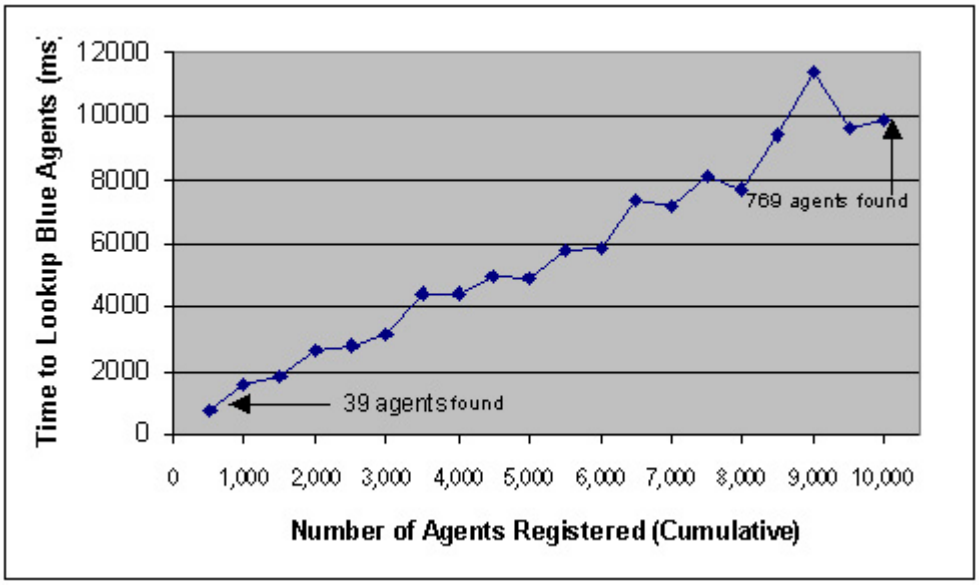

Figure 23. Looking Up Multiple (Blue) Agents [49]

Figure 24 shows a normalized graph that indicates that the lookup of multiple agents is roughly proportional to the number of agents retrieved.

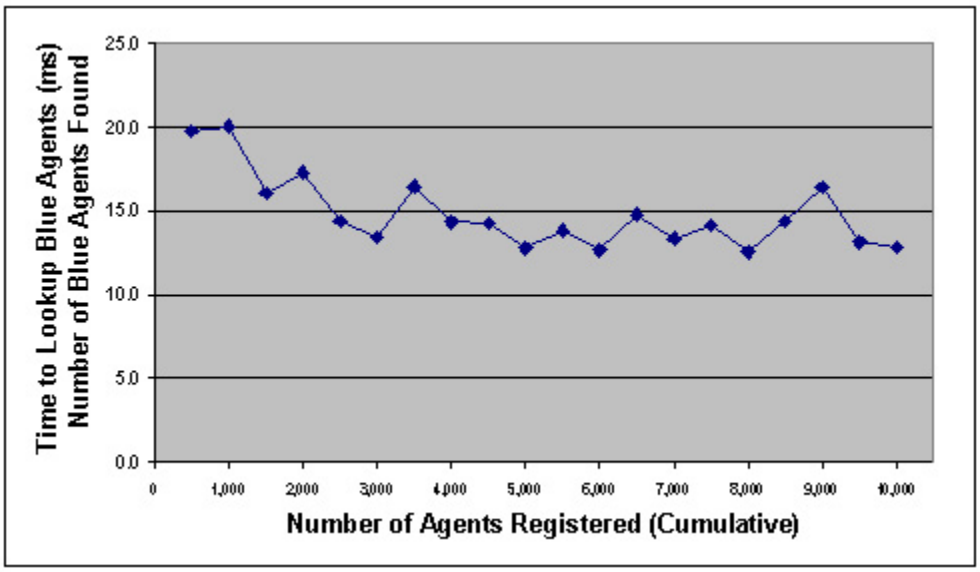

Figure 24. Normalized Looking Up Multiple (Blue) Agents [49]

In April 2001, a second set of experiments were run with a newer version of the Grid code. These experiments repeated the lookup experiments done previously.

Measurements of how long it took to register the agents, as the lookup service became more populated were also collected. Figure 25 shows the registration time results [51]:

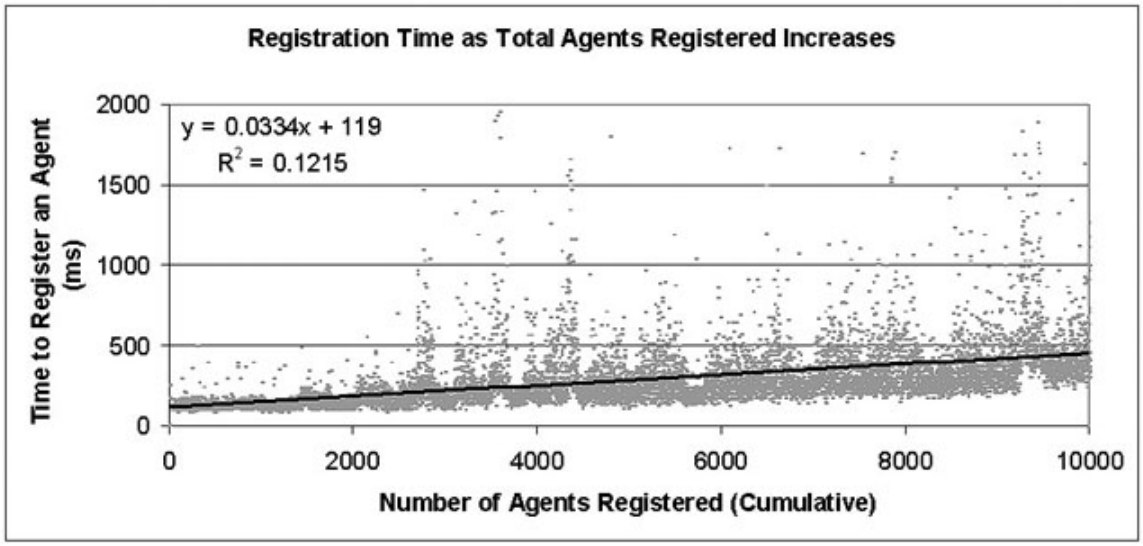

Figure 25. Results from April 2001 experiment [51]

B. CONCEPTUAL APPLICATION FOR ENTERPRISE NETWORK MANAGEMENT IN THE ARMY

Having discussed the requirements and issues of enterprise network management in the Army, and looking at the many capabilities that intelligent agent technology and multi-agent systems brings, this section discusses the conceptual application of these technologies for the Army Enterprise Infostructure initiative. This section describes a high-level enterprise network management architecture by integrating the concepts of an intelligent-agent-based architecture [10], designed by SRI International, on top of the CoABS multi-agent environment. The idea here is just to discuss an approach that emphasizes the potential of these technologies to enhance the Army's efforts. This architecture is by no means the best approach or optimal intelligent-agent-based solution; there are many other ways to exploit these and other agent-based technologies.

While CoABS has shown promising results for the interoperability of command and controls systems within in the military, this technology has not been applied to the NM domain. CoABS is ideal for enhancing NM capabilities. As discussed in the previous section, the CoABS Grid demonstrates the rich features (such as scalability, reliability, flexibility, and adaptability) that are necessary for enterprise network management in the

Army. The Grid is capable of internetworking heterogeneous intelligent agent NM platforms in a fashion analogous to routers internetworking heterogeneous local area networks. Such an environment allows the enterprise to monitor and control the network in an efficient, effective, and low-cost manner.

1. The Intelligent-Agent-Based Enterprise Management Architecture

The agent architecture described in this section is integrated on top of CoABS to allow for agent communication, functionality, and services, and for central enterprise management and control of the network. It is based on a federated and hierarchical design where agents communicate, coordinate, and collaborate over the CoABS infrastructure, but implements control of the network nodes in a hierarchical manner. CoABS facilitates interoperability of legacy systems, proprietary agent environments that are Grid-aware, and other disparate systems that are Grid-aware. The architecture also provides the necessary capability to aggregate NM situational awareness information into the enterprise NM common operational picture (see Fig. 26).

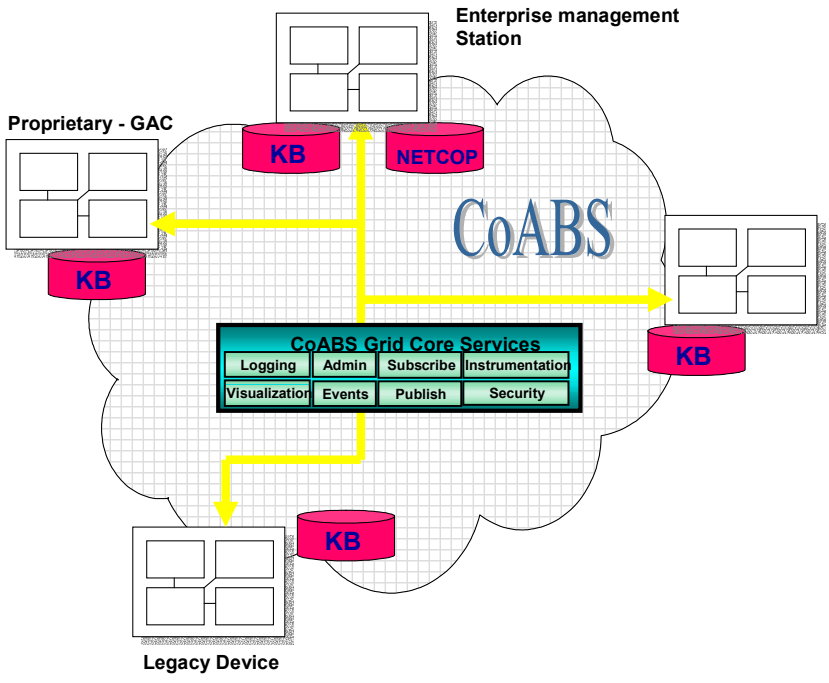

Figure 26. Intelligent-Agent-Based High-level Architecture

The architectural approach [10] is based on an infrastructure in which data-driven processes (i.e., intelligent local agents) can analyze network conditions and post the results to a shared memory structure. Each of the agents represents a different technology, and is activated upon receiving the appropriate data or information. The advantage of this approach is that when a new method of analyzing data is developed, that method can be incorporated into the architecture as a new agent. The adjustments are then limited to registering that agent and giving the agent the ability to produce and evaluate agent-to-agent messages. Thus, the overhead for changes is much less than with traditional approaches.

As shown in Figure 27, a set of localized agents are embedded within each managed network node to perform specific management tasks. Separate agents exist to carryout tasks for a specific NM function: fault, configuration, accounting, performance, or security management. Only performance and fault management are addressed in this architecture, although the model can incorporate others. With this design, intelligent NM functionality is pushed out to the devices where autonomous troubleshooting, computational processing, and repair can occur dynamically. Agents perform their NM functions within a network node by analyzing data as it arrives and posting results or conclusions to a shared memory. While it is processing data, each agent has access to the shared memory and the conclusions of other agents.

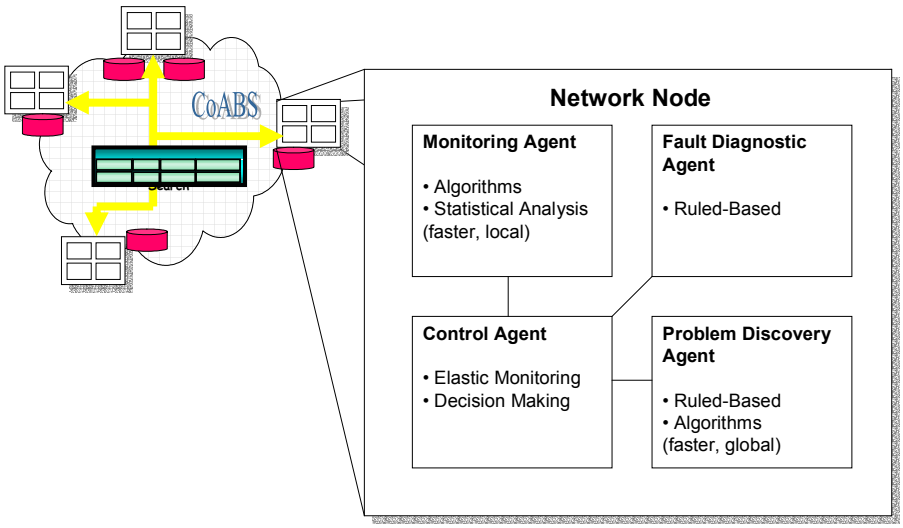

Figure 27. The Intelligent-Agent-Based Network Node Architecture

The agent architecture consists of the following four functional intelligent agents that perform very different tasks (see Fig. 27) [10]:

- **Control Agent** – uses symbolic and analogical reasoning to synthesize results from the other (local and peer-to-peer) agents, additional network information, and information from other nodes, and to make decisions based on QoS considerations.
- **Fault-Diagnostic Agent** - uses rules to isolate faults in the network. This agent usually works over a longer timeframe than other agents. The incoming data typically identifies deviant network behavior (originating from the network) or potential hot spot areas (originating from other agents), and the problem-solving process of fault isolation and recovery may not occur in real time.
- **Monitoring Agent** - uses statistical and algorithmic approaches to analyze information from all levels of network performance, in real time. The Monitoring Agent is interested in collecting situational awareness data originating from the network and in real-time analyses of that data (e.g., congestion detection).
- **Problem Discovery Agent** - uses rules and algorithms to isolate geographic problem areas in the network that need further investigation, in a short timeframe. The Problem Discovery Agent is interested in

discovering problem areas in a short time frame and in avoiding transient problems. The information analyzed by this agent comes directly from the network.

The Control Agent serves as the decision maker, ensuring that the local agents work together towards a common goal. It synthesizes results and data reported by the other local agents and the information passed on by other nearest-neighbor Control Agents and then makes control decisions to optimize the QoS of the network. The Control Agent thus takes a local view within each node, but also considers the more global view of the network through communication with its nearest neighbor nodes. The Control Agent correlates the results of all the local agents, receives requests from the local agents for different data or monitoring needs, requests data from other Control Agents of other nodes (i.e., peer-to-peer or global communication), and makes intelligent control decisions for optimizing the network performance.

Within each local node, the agents are not aware of the other agents in the node. They are only aware of the data that is available in the shared memory and the data passes to them by the Control Agent. This design approach was implemented to reduce the overhead within the nodes created by agent-to-agent communication messages.

The Control Agent can perform what is known as "elastic monitoring," [10] where the monitoring of the network is varied according to the current network state (i.e., congestion and other factors). In this manner, care is taken to avoid monitoring at levels that would exacerbate any network problems, as is the case with traditional models.

2. Knowledgebase

Instead of storing management information within the traditional management information base (MIB) as in the SNMP model, this architecture incorporates semantically designed knowledgebase's. The advantage of a knowledgebase is that they give agents the capability to reason and infer for intelligent decision-making. Extensible Mark-up Language-based (XML-based) technologies can be used to structure the knowledgebase, thus making it semantically enabled for agents to understand. For instance, the knowledgebase structure can be implemented with the OWL Web Ontology

Language [52] that creates a marked-up data structure schema where agents can understand and process the content of information and reason and infer based on the ontology structural relationships.

For legacy systems that are incompatible, XML-based technologies can be used to wrap the database and extract the contextual management information. The contextual information is then stored in the knowledgebase.

The knowledgebase also incorporates the shared memory where the various agents deposit critical information for future reference. This provides a resource for agents to learn from past experience. For example, the Fault Diagnostic agent can query the knowledgebase to determine recovery procedures captured in the shared memory.

The knowledgebase allows the Control Agent to learn from past experience, reason, and determine the best control action for the network autonomously. Additionally, the Control Agent correlates the analyses posted in the shared memory by the other agents. This capability tremendously improves network management, especially at the enterprise-level where a large part of the burden is now distributed across the network. Of course, the human administrator sets and controls the degree of autonomous agent actions.

3. Integration with CoABS Grid

For integration of the agent architecture on to the CoABS Grid, the CoABS middleware application is loaded on the network nodes. This enables the local device agents to be Grid-aware and interface with the CoABS Grid services. This allows the agents to communicate and collaborate over the Grid. Messages and alarms are forwarded to the enterprise management entity over the Grid. Additionally, the enterprise entity can launch agents over the Grid query selected devices on an as needed basis.

4. Network COP

Situational awareness of the network is captured by the Monitoring Agent on each of the nodes. The nodes only collect and store situational awareness information that is pertinent to itself. The Control Agents at the nodes retrieve situational awareness information and reports it to the enterprise NETCOP as events occur. Situational awareness information is stored in a NETCOP knowledgebase where the Monitoring

Agent in the management station aggregates the situational awareness information and displays it for administrator review. The Control Agent in the management station can analyze information in the NETCOP knowledgebase and provide recommendations for decision support.

5. Example

Here is a short example that sums up the functionality of the agent architecture [10]:

> Suppose that the Monitoring Agent has detected and reported congestion in a general area of the network. This is reported to the Control Agent, which then retrieves cases analogous to the problem description. On the analogy of one such case, the Control Agent decides to ask the Problem Detection Agent to locate the problem area to a finer granularity, and to ask the Monitoring Agent to increase monitoring in that area of the network without substantially increasing the load on the network. The Control Agent also asks its nearest neighbor nodes if they are seeing trouble in the same area and if they have come to any conclusions as to the cause. In the meantime, the Problem Discovery Agent has narrowed down the problem to an area of the network that is not responding. This implies a fault, so the Control Agent issues a request to the Fault Diagnostic Agent to isolate the fault. The Fault Diagnostic Agent uses the shared memory data and issues requests for additional monitoring if needed, while performing its fault isolation activities. Once the fault is isolated, the Fault Diagnostic Agent posts it on the shared memory and announces the event to the Control Agent. The Control Agent then notifies its nearest neighbor nodes of its conclusions.

6. Closing Comments

Relative to the traditional models, the implications of the design described above are increased reliability because problems are isolated at the device and agent action is no longer at the mercy of the network, improved adaptability (and reliability) because the agents have the intelligence to respond to changing conditions, and increased flexibility because the agents are modular and have specific independent responsibilities. Network management now becomes fully distributed while still maintaining central control and management of the entire network from a central location. These properties are also inherent in the CoABS Grid. The CoABS platform provides scalability, which is essential for the AEI. CoABS provides this scalability without constraining network resources as with the traditional protocols.

THIS PAGE INTENTIONALLY LEFT BLANK

VII. CONCLUSION AND RECOMMENDATION FOR FUTURE RESEARCH

A. CONCLUSION

This study concludes that in theory, intelligent-agent-based technologies are a leading solution, among other current technologies, to achieve the Army's enterprise network management goals for the AEI. The research brought out and discussed the many shortcomings of the traditional protocols such as SNMP that will hinder AEI implementation. It elaborated on the robust capabilities of intelligent agents and multi-agents systems and how they can be applied to mitigate many of the SNMP shortfalls. An argument was made that demonstrated the advantages of intelligent-agent-based NM over SNMP and other protocols. The study further reviewed the DoD's overarching JV2010/2020 efforts and the DoD's and the Army's approaches to establishing enterprise NM systems. Finally, the study presented a conceptual architecture that showed how intelligent-agent-based technologies can be applied to achieve the Army's AEI objectives.

The research and analysis discussed and answered the research questions as follows:

1. What alternative technologies can scale and meet the Army Enterprise Infostructure (AEI) network management and situational awareness requirements?

As mentioned throughout this paper, intelligent-agent-based technologies offer the scalability, flexibility, reliability, and adaptability needed for the AEI and situational awareness requirements. These characteristics were illustrated in the conceptual architecture where an agent-based model was placed on top of the CoABS Grid to create the dynamic environment. Although intelligent agent technologies is a long way off, it is an alternative that the Army should study for future use as the JV2010/2020 evolves.

Sub-research questions:

a. **How can intelligent-agent-based technologies be used to establish network management control and network situational awareness of the AEI?**

As discussed in Chapter III, intelligent agents can assume many properties that allow them to dynamically manage and control a network environment. The multi-agent system provides the infrastructure where agents can communicate and collaborate to collectively solve NM problems as needed. This was also discussed in the conceptual architecture where intelligence was pushed out to the managed devices where NM could occur autonomously.

b. How can intelligent-agent-based technologies be used to execute Fault, Configuration, Accounting, Performance, and Security (FCAPS) management or establish a network common operational picture (NETCOP)?

The was depicted in the conceptual architecture where the managed devices consists of various intelligent agents to perform FCAP functions and report only distilled knowledge to the enterprise operations. The NETCOP is aggregated by the device control agents reporting to the enterprise NETCOP knowledgebase as the situation dictates.

c. How does intelligent-agent-based technology for enterprise network management compare to SNMP-based and other distributed management technologies?

This question was answered in Chapter II where the argument for intelligent agents in NM was made. The Lucent experiments demonstrated that an intelligent-agent-based platform was more dynamic, could handle multiple standards, overcame interoperability problems, is inherently distributed, and demonstrated the potential for much greater cost-savings than the traditional protocols.

d. Can an intelligent-agent-based network management architecture scale to support the AEI?

The CoABS experiments found that sequential lookup scales well to 10,000 agents. Lookups that retrieved a single agent, as well as lookups that matched no agents were not affected by the number of agents registered. Additionally,

time to lookup multiple agents increased proportionally to the number of agents retrieved. Finally, time to lookup multiple agents is independent of the number of agents registered.

e. How can the Control of Agent Based Systems (CoABS) be leveraged to support an intelligent-agent-based enterprise-level network management architecture for the AEI?

As discussed in Chapter VI, the CoABS Grid is a framework for federating heterogeneous agent systems designed to meet the challenges of the military environment, as well as address the heterogeneity among the participating agent research communities. The CoABS Grid is an adaptive and robust collection of infrastructure, services, agents, standards, and protocols. It enables the run-time integration and dynamic interoperability of distributed agents, objects, devices, and legacy systems. As explained in the conceptual architecture, CoABS serves as the infrastructure that allows the agents to accomplish their goals.

B. RECOMMENDATION FOR FUTURE RESEARCH

This study gives a comprehensive review of how agent-based technologies can be used to solve the enterprise NM issues. However, this area certainly merits much more research and application. The Army should investigate the issues and the potential of intelligent-agent-based technologies with respect to the AEI implementation. A technical solution should be pursued and demonstrated with a small-scale proof-of-concept prototype. The technologies in this study are just some of the readily available and accessible technologies. The Army should conduct an all-out research effort, on and off the commercial market, to find the most optimal and mature technologies that meet the Army's requirements.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX I NETWORK MANAGEMENT TECHNOLOGIES

A. INTRODUCTION

To make the argument for intelligent-agent-based technologies, it is important to understand the functionality of the current technologies in order to assess their limitations. This section provides an overview of the most common technologies in use today with emphasis and a more detailed review of the SNMP, which is the most prevalent.

B. SIMPLE NETWORK MANAGEMENT PROTOCOL (SNMP)

Since its inception, SNMP evolved in three different versions (SNMPv1, SNMPv2, and SNMPv3) that provide increased functionality to overcome some of the design inherent weaknesses. SNMPv1 was too simplified, meaning its simplicity and the lightweight nature of the agent only allowed device status report and update, while the burden of management and data processing resided with the manager. Version 1 severely also lacked security to protect it from internet sabotage. SNMPv2 introduced the concept of intermediary manager [29] or "middle manager." The intermediary managers decentralized management responsibility by assuming some of the data processing from the manager side. Additionally, the intermediaries are capable of performing simple tasks. SNMPv2 also added bulk transfer of information capabilities and other functional extensions, but still lacked the necessary security. In 1998, the most recent version, SNMPv3, was issued. SNMPv3 provides the much needed security features of authentication, privacy and access control. It also provides other enhancements such as modularity, which allows for module upgrades without needing to issue a new entire standard.

SNMP is based on three concepts [27]: managers, agents, and the Management Information Base (MIB). In any configuration, at least one manager node, called the network management station (NMS), runs SNMP management software. Network devices to be managed are commonly called network nodes or network elements (NE). Devices such as bridges, routers, servers, and workstations, are equipped with an agent software module. The agent is responsible for providing access to a local MIB of objects that reflects the resources and activity at its node. The agent also responds to manager

commands to retrieve values from the MIB and to set values in the MIB. The MIB is in essence a database schema for storing management data, which are called managed objects. An example of an object that can be retrieved is a counter that keeps track of the number of packets sent and received over a link into the node; the manager can track this value to monitor the load at that point in the network. An example of an object that can be set is one that represents the state of a link; the manager could disable the link by setting the value of the corresponding object to the disabled state. Many system software vendors include SNMP manager and agent programs as standard software components. It is unusual nowadays to have to write them.

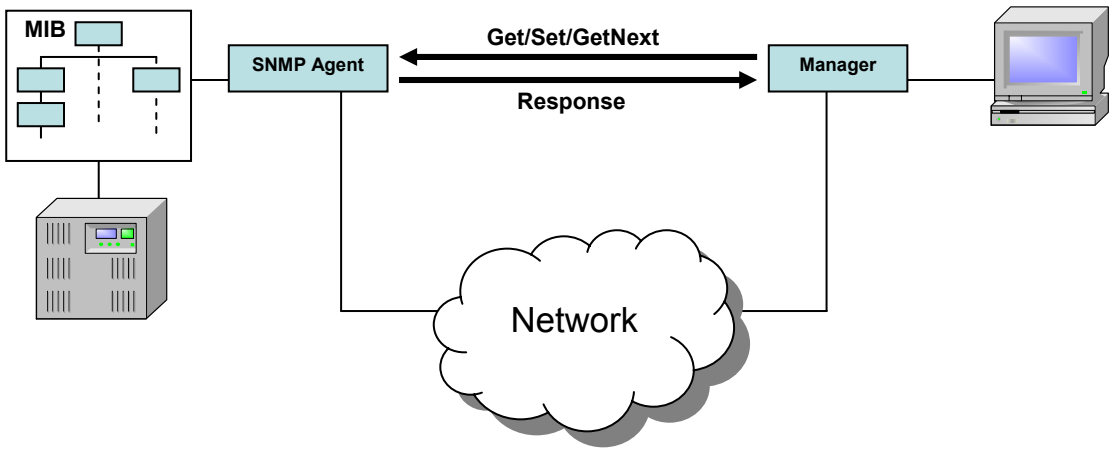

Figure 28. SNMP Components

1. The SNMP Agent

SNMP agents [28] (see Fig 28) reside in the managed network and communicate with network management stations. SNMP agents are embedded on managed devices. SNMP agents provide the following functionality:

- Implementing and maintaining MIB objects
- Responding to management operations such as requests
- Generating notifications, both traps (unacknowledged) and informs (acknowledged)
- Setting the access policy for external managers

- Implementing security—SNMPv1 and SNMPv2c support community-based security with clear-text passwords; stronger security (authentication and encryption) is available with SNMPv3. SNMPv3 also provides an access control framework, which consists of:

    - ◊ MIB view—the set of managed objects in an agent MIB accessible to an SNMP manager. This is the manager's client view with respect to the agent.

    - ◊ Access mode to managed objects—either READ-ONLY or READWRITE. A READ-ONLY access mode means that no agent MIB objects can be written by a manager. MIB views are associated with specific access modes.

SNMP agents can be hosted on almost any computing device, including: Windows NT/2000 machines, UNIX hosts, Novell NetWare workstations and servers, and many network devices, including hubs, routers, switches, terminal servers, PABXs, and so on.

The agent listens on the managed device's User Datagram Protocol (UDP) port 161 for the following SNMP message types:

- **Get** requests the values of the specified object instances.
- **Get-next** requests the values of the lexical successors of the specified object instances.
- **Get-bulk** requests the values of portions of a table.
- **Set** modifies a specified set of object instance values.

The above messages either retrieve (get) or modify (set) NE data as defined in the MIB. The agent uses UDP port 162 for sending notification messages to a preconfigured IP address.

2. The SNMP Manager

SNMP managers (NMSs) [28], shown in Figure 28, are the entities that interact with agents. Their primary functions are: getting and setting the values of MIB object instances on agents, receiving notifications from agents, and exchanging messages with other managers.

3. The MIB

A MIB [28] is simply a managed-object data description. The MIB defines the syntax (type and structure) and semantics of the managed objects. SNMP managers and agents exchange managed object instances using the SNMP protocol. Managed objects may be defined using what are called textual conventions. These are essentially refinements of basic types (that are very loosely analogous to programming language data types or even Java/C++ classes). Some of the textual conventions are:

- **MacAddress** is an IEEE 802 MAC address.
- **TruthValue** is a boolean value representing true (1) or false (2).
- **TestAndIncr** prevents two managers from simultaneously modifying the same object. Setting an object of type TestAndIncr to a value other than its current value fails. We will see a similar mechanism used in the MPLS tables.
- **RowStatus** is a standard way for adding and removing entries from a table (we will see this object used many times in the MPLS configuration examples).
- **StorageType** specifies how a row should be stored.

In addition to using textual conventions, MIB objects have common attributes. Managers use these attributes in order to manipulate and understand MIB objects. Below is a list of common attributes:

:

- **SYNTAX**: This is the object format—for example, Unsigned32 (an integer), TruthValue (a Boolean true or false), and SEQUENCE (a container of other objects).
- **MAX-ACCESS**: This specifies the accessibility of the object—for example, read-only means that the object can only be read (but not written) by managers.
- **STATUS**: This is the state of support for the object in the MIB—for example, current means that the object is relevant and can or should be supported.
- **DESCRIPTION**: This is a text description of the object.

- **DEFVAL**: This is a default value that the agent can use when the object instance is first created.

- **OBJECT IDENTIFIER**: This is the unique name for a MIB object, described in the next section.

OIDs and Lexigraphical Ordering. All MIB objects have unique names called object identifiers (OIDs) [28]. An OID is a sequence of 32-bit unsigned integers that represents a node within a tree-based structure (with a single root). Figure 29 shows the basic construct of a MIB Tree with assigned OIDs on each node. Only an instance of a MIB object can be retrieved from an agent. An instance of a MIB object is identified by an OID concatenated with the instance value. The instance value is a sequence of one or more 32-bit unsigned integers.

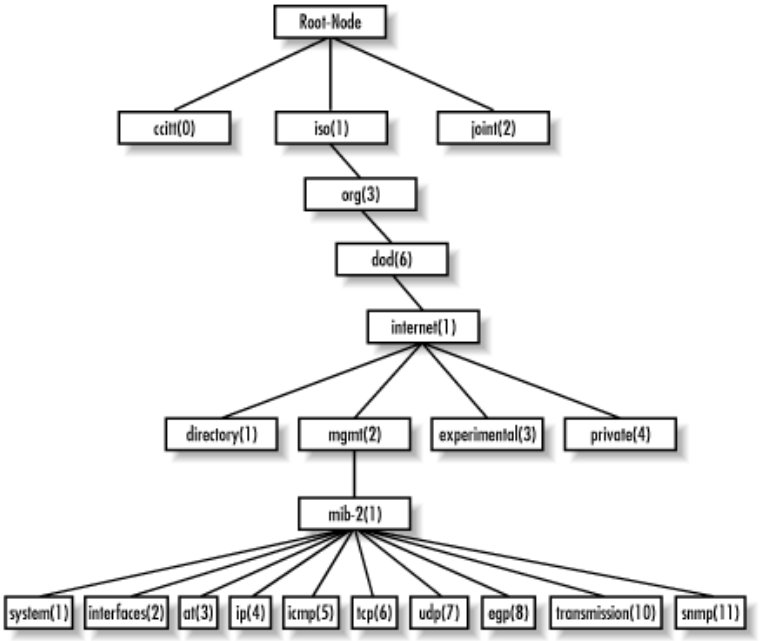

Figure 29. Sample MIB Tree [31]

The order of the OIDs, called "lexigraphical ordering," [28] is an important aspect of SNMP. All objects can be traced from the root in a process called "walking the MIB." During a walk, each branch of the MIB tree is traversed from left to right starting at the root. For example, the standard IP group or table has the OID 1.3.6.1.2.1.4, as illustrated in Figure 30. The IP group and some of its constituent objects are shown in this diagram [28].

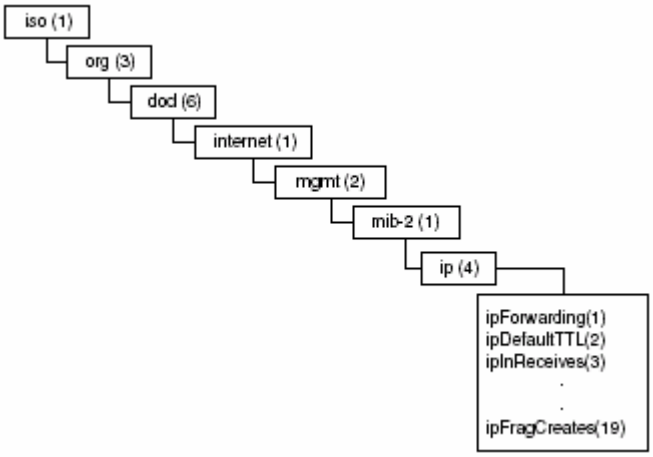

Figure 30. The MIB-II IP Group [28]

MIBs are plain-text files. They are compiled into the agent source code and become part of the executable file. If a manager wants to access some agent MIB objects, then either the associated MIB module file is needed or a MIB walk can be attempted.

Another important aspect of lexicographic ordering is that a manager can use it to "discover" an agent MIB. This is for that case in which the manager does not have a copy of the agent MIB and needs to determine what objects the agent supports. The discovery process consists of walking the MIB. It should be noted that this is not a very good way of retrieving agent data. It is far better to have the MIB details at the manager side because the structure and meaning of the NE data will then be apparent.

4. SNMP Protocol Data Unit (PDU)

SNMP managers and agents communicate using a very simple messaging protocol. This is a straightforward fetch (get), store (set), and notification model. Managers retrieve agent data using get operations, and they modify agent data using set operations. When agents want to communicate some important event, they do so by sending a notification message to a preconfigured IP address. If the agent wants to receive an acknowledgment from the manager, then it sends an inform message. Below is a brief overview of the SNMP message formats [32].

Figure 31. SNMVPv1 message format [32]

SNMPv1. SNMPv1 messages contain two parts: a message header and a protocol data unit (PDU) [28]. Figure 31 illustrates the basic format of a SNMPv1 message. SNMPv1 message headers contain two fields: Version Number and Community Name. The following descriptions summarize these fields:

- **Version number**—specifies the version of SNMP used.
- **Community name**—defines an access environment for a group of NMSs. NMSs within the community are said to exist within the same administrative domain. Community names serve as a weak form of authentication because devices that do not know the proper community name are precluded from SNMP operations.

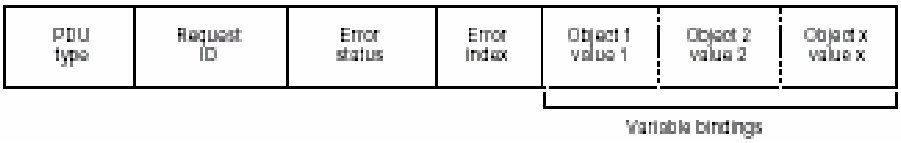

Figure 32. SNMPv1 PDU [32]

SNMPv1 PDUs contain a specific command (Get, Set, and others) and operands that indicate the object instances involved in the transaction (see Fig 28). The SNMPv1 Get, GetNext, Response, and Set PDUs contain the same fields. Note that SNMPv1 PDU fields are variable in length. The following descriptions summarize the fields illustrated in Figure 32:

- **PDU type**—specifies the type of PDU transmitted.
- **Request ID**—associates SNMP requests with responses.
- **Error status**—indicates one of a number of errors and error types. Only the response operation sets this field. Other operations set this field to zero.
- **Error index**—associates an error with a particular object instance. Only the response operation sets this field. Other operations set this field to zero.
- **Variable bindings**—Serves as the data field of the SNMPv1 PDU. Each variable binding associates a particular object instance with

its current value (with the exception of Get and GetNext requests, for which the value is ignored).

Figure 33. SNMPv1 Trap PDU [32]

Figure 33 shows the message format for a SNMPv1 Trap message. The following descriptions provide a summary of the fields:

- **Enterprise**—identifies the type of managed object generating the trap.
- **Agent address**—provides the address of the managed object generating the trap.
- **Generic trap type**—indicates one of a number of generic trap types.
- **Specific trap code**—indicates one of a number of specific trap codes.
- **Time stamp**—provides the amount of time that has elapsed between the last network reinitialization and generation of the trap.
- **Variable bindings**—the data field of the SNMPv1 Trap PDU. Each variable binding associates a particular object instance with its current value.

SNMPv2. The Get, GetNext, and Set operations used in SNMPv1 are exactly the same as those used in SNMPv2. However, SNMPv2 adds and enhances some protocol operations. The SNMPv2 Trap operation, for example, serves the same function as that used in SNMPv1, but it uses a different message format and is designed to replace the SNMPv1 Trap.

SNMPv2 also defines two new protocol operations: GetBulk and Inform. The GetBulk operation is used by the NMS to efficiently retrieve large blocks of data, such as multiple rows in a table. GetBulk fills a response message with as much of the requested

data as will fit. The Inform operation allows one NMS to send trap information to another NMS and to then receive a response. In SNMPv2, if the agent responding to GetBulk operations cannot provide values for all the variables in a list, it provides partial results.

Figure 34. SNMPv2 GetBulk PDU [32]

The SNMPv2 GetBulk PDU Consists of Seven Fields, as shown in Figure 34. The following descriptions summarize the GetBulk fields:

- **PDU type**—identifies the PDU as a GetBulk operation.
- **Request ID**—associates SNMP requests with responses.
- **Non repeaters**—specifies the number of object instances in the variable bindings field that should be retrieved no more than once from the beginning of the request. This field is used when some of the instances are scalar objects with only one variable.
- **Max repetitions**—defines the maximum number of times that other variables beyond those specified by the Non repeaters field should be retrieved.
- **Variable bindings**—Serves as the data field of the SNMPv2 PDU. Each variable binding associates a particular object instance with its current value (with the exception of Get and GetNext requests, for which the value is ignored).

SNMPv3. SNMPv3 [33] is designed to be backward compatible with SNMP versions 1 and 2 and add security in the form of access control, authentication, and encryption to existing SNMP implementations. As such, version 3 is essentially version 2 with the addition of security features and other enhancements. Two of the most significant additions provided by SNMPv3 are the User-based Security Model (USM) and View-based Access Control Model (VACM).

The User-based Security Model (USM) of SNMPv3 defines mechanisms for providing message-level security for SNMP implementations. The USM is designed to protect against threats such as:

- **Modification of information** – changing management information in transit between the SNMP manager and agent
- **Masquerade** – a non-authorized user assuming the identity of a user authorized to perform management operations
- **Message stream modification** – reordering or copying packets in a management message stream for malicious purposes
- **Disclosure** – a non-authorized user accessing a message in transit to learn information (e.g., passwords) contained in the stream

SNMPv3 provides authentication, ensures data integrity, and prevents masquerading. After a network manager logs on to a management station with a username and password, SNMPv3 authentication consists of applying MD5 (Message Digest 5) or SHA (Secure Hash Algorithm) to PDU packets using a key. The algorithm produces an authentication value and places it in the message. The receiver applies the same algorithm with the same key and checks if its produced value is the same as the one in the message. The key used for the authentication is associated to the network manager's user name, which is present within the SNMP message. The authentication functionality ensures that each SNMP message comes from an authorized manager or agent and that it was not tampered with in transit.

The USM also has mechanisms for checking the timeliness of SNMP PDU delivery using synchronization and time-window checking techniques. This helps detect messages that have been delayed, which is important because delay is often an indicator that packets have been altered.

The SNMPv3 View-based Access Control Model (VACM) is designed to control access to management information based on a user's identity. The VACM allows different access levels (read, write, notify) to be defined for different users and for each piece of MIB information. After a network manager authenticates as specified in the USM, all SNMP commands generated carry his/her credentials. SNMP agents check the

user's information against a pre-configured access control database before allowing access to any MIB object. This gives network managers the ability to define different access rights for different administrators.

With all the added features, the SNMPv3 message format is significantly different than its predecessors. The SNMPv3 message structure is illustrated in Figure 35 [33].

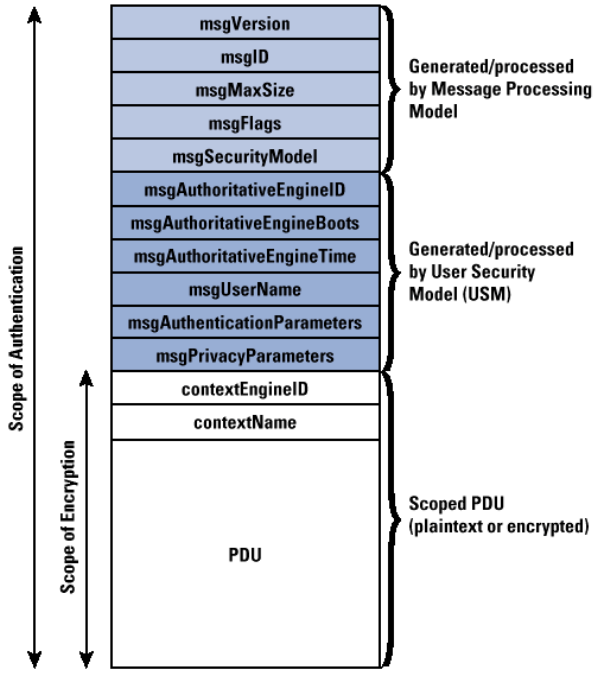

Figure 35. SNMPv3 message format with USM [33]

The first five fields are generated by the message processing model on outgoing messages and processed by the message processing model on incoming messages. The next six fields show security parameters used by the security model, which is invoked by the message processing model to provide security services. Finally, the PDU, together with the contextEngineID and contextName, constitute a scoped PDU, used for PDU processing. The first five fields follow:

- **msgVersion**: Set to snmpv3
- **msgID**: A unique identifier used between two SNMP entities to coordinate request and response messages, and by the message processor to coordinate the processing of the message by different subsystem models within the architecture

- **msgMaxSize**: Conveys the maximum size of a message in octets supported by the sender of the message

- **msgFlags**: An octet string containing three flags in the least significant three bits: reportableFlag, privFlag, authFlag.

- **msgSecurityModel**: An identifier that indicates which security model was used by the sender to prepare this message

To summarize the different PDUs for each of the versions, Table 2 below displays a PDU taxonomy.

SNMPv1	SNMPv2c	SNMPv3	Response PDU
GetRequest	GetRequest	GetRequest	GetResponse
GetNextRequest	GetNextRequest	GetNextRequest	GetResponse
SetRequest	SetRequest	SetRequest	GetResponse
Trap	Trap	Trap	None
	GetBulkRequest	GetBulkRequest	GetResponse
	InformRequest	InformRequest	GetResponse

Table 2 Protocol Data Units in the Different Versions of SNMP [28]

C. COMMON MANAGEMENT INFORMATION PROTOCOL (CMIP)

Common Management Information Protocol (CMIP) [30] is an Open Systems Interconnection (OSI) -based network management protocol that supports information exchange between network management applications and management agents. Its design is similar to the Simple Network Management Protocol (SNMP). CMIP was developed and funded by government and corporations to replace and makeup for the deficiencies in SNMP, thus improving the capabilities of network management systems.

CMIP does not specify the functionality of the network management application, it only defines the information exchange mechanism of the managed objects and not how the information is to be used or interpreted. It uses an ISO reliable connection-oriented transport mechanism and has built in security that supports access control, authorization and security logs.

Communication between the agents and the management application is accomplished through objects. These are the same managed objects described in the SNMP overview. The network management application can initiate transactions with management agents using the following operations:

- **ACTION** - Request an action to occur as defined by the managed object.
- **CANCEL_GET** - Cancel an outstanding GET request.
- **CREATE** - Create an instance of a managed object.
- **DELETE** - Delete an instance of a managed object.
- **GET** - Request the value of a managed object instance.
- **SET** - Set the value of a managed object instance.

CMIP was designed to be more robust and efficient than SNMP. Here are some of the built-in advantages:

- CMIP is a safer system as it has built in security that supports authorization, access control, and security logs.
- CMIP provides powerful capabilities that allow management applications to accomplish more with a single request.
- CMIP provides better reporting of unusual network conditions

While CMIP was destined to replace SNMP, it was never fully embraced because of several limitations that surfaced as a result of the design and universal acceptance. Here are a few limitations that stand out:

- CMIP is widely used in the telecommunication domain and telecommunication devices typically support CMIP.
- The CMIP protocol is designed to run on the ISO protocol stack. However, the technology standard used today in most LAN environments is TCP/IP and most LAN devices only support SNMP.
- CMIP requires a large amount of system resources; this has resulted in very few implementations. Additionally, CMIP is very complex thus making it difficult to program; therefore skilled

personnel with specialized training may be required to deploy, maintain and operate a CMIP based network management system.

D. REMOTE MONITORING (RMON)

In 1995, the Internet Engineering Task Force (IETF) developed a decentralized NM paradigm called Remote Monitoring (RMON) [29]. RMON uses the concept of monitors or probes that provides analysis of network traffic, as opposed to the devices with ordinary SNMP agents. Probe implementation can be done as device embedded applications or as separate devices. The task of a probe is to monitor the network traffic at its local region and report anomalies, in the form of alarms, to its manager. By defining alarm types and alarm thresholds, the manager is able to offload some data gathering and decision-making (mainly event filtering) to the probes. Furthermore, the probes can also perform some data pre-processing before forwarding them to the manager. In general, the earlier works towards distributed network management can be considered as weak distribution. The management tasks still reside heavily on the manager side, and some rudimentary management duties are delegated to intermediary entities, in the form of event filtering, notification, and data pre-processing.

E. COMMON OBJECT REQUEST BROKER ARCHITECTURE (CORBA)

The Common Object Request Broker Architecture (CORBA) [30] is a specification of a standard architecture for object request brokers (ORBs). An ORB is a middleware technology that manages communication and data exchange between objects. ORBs promote interoperability of distributed object systems because they enable users to build systems by piecing together objects from different vendors that communicate with each other via the ORB. Thus, vendors can develop ORB products that support application portability and interoperability across different programming languages, hardware platforms, operating systems, and ORB implementations.

Using a CORBA-compliant ORB, a client can transparently invoke a method on a server object, which can be on the same machine or across a network. The ORB intercepts the call, and is responsible for finding an object that can implement the request, passing it the parameters, invoking its method, and returning the results of the invocation. The client does not have to be aware of where the object is located, its programming language, its operating system or any other aspects that are not part of an object's

interface. The vision behind CORBA is that distributed systems are conceived and implemented as distributed objects. The interfaces to these objects are described in a high-level, architecture-neutral specification language that also supports object-oriented design abstraction.

The CORBA specification was developed by the Object Management Group (OMG), an industry group with over six hundred member companies representing computer manufacturers, independent software vendors, and a variety of government and academic organizations. Thus, CORBA specifies an industry/consortium standard, not a formal standard in the IEEE/ANSI/ISO sense of the term. The OMG was established in 1988, and the initial CORBA specification emerged in 1992. Since then, the CORBA specification has undergone significant revision, with the latest major revision (CORBA v2.0) released in July 1996

Benefits: CORBA works with SNMP, CMIP and most major element management and network management platforms. To simplify the migration from SNMP and CMIP to CORBA, the TeleManagement Forum (TMF) and OMG have defined standard mappings between the information models and protocols.

Most major carriers are already using and promoting CORBA. Because CORBA is pervasive in enterprise information technology applications, it is already heavily used by telecommunications service providers. In addition, CORBA is the means that most element management platforms and network management platforms use to communicate with each other. By extending the use of CORBA to Network Element device management, carriers can simplify their network management systems and more tightly integrate their management and IT networks.

Limitations:

- Programming language support. IDL is a "least-common denominator" language. It does not fully exploit the capabilities of programming languages to which it is mapped, especially where the definition of abstract types is concerned.

- Pricing and licensing. The price of ORBs varies greatly, from a few hundred to several thousand dollars. Licensing schemes also vary.

- Training. Training is essential for the already experienced programmer: five days of hands-on training for CORBA programming fundamentals is suggested.

- Security. CORBA specifies only a minimal range of security mechanisms; more ambitious and comprehensive mechanisms have not yet been adopted by the OMG.

F. OTHER TECHNOLOGIES

The market consists of many non-open standards NM solutions as well. The next few paragraphs summarize other NM technologies that are proprietary [28]:

Microsoft Systems Management Server (SMS) allows system administrators very flexible control of networks of Windows machines. Software applications deployed on host machines can be determined by remotely viewing the local Windows registry (a type of configuration database) on each machine. This can be very useful for verifying on large sites that software licenses have not been exceeded—too many users installing a given package. SMS also allows software to be distributed to destination machines. A major drawback of SMS is that it only works on Windows machines. In other words, it is technology dependent, unlike open standard technologies such as SNMP which is technology neutral. Therefore, SMS is a solution for select organizations that require this capability.

Telnet refers to a menu-based Command Line Interface (CLI) style of management. This approach requires the management user to connect to the IP address of a given device using telnet. The device then provides a text menu-based application with which the user interacts. This is useful and is a widely adopted approach for device management. It is generally possible to use telnet to configure devices such as laser printers, routers, switches, and terminal servers. The problem with it is that menu-based management systems are proprietary by their nature and don't easily lend themselves to centralized, standards-based management (as does SNMP).

Serial link-based menu systems are very similar to NEs that support telnet. Just the access technology is different. Normally, a serial link-based system includes simple text menus (accessed via a serial interface) that are used for initial configuration. Typical devices for these facilities include small terminal servers. Often, these devices do not

have an IP address, and the user configures one via the menu system. Connecting the device to an appropriately configured PC serial port facilitates this. Again, by its nature this is proprietary.

Desktop Management Interface (DMI) was developed by the Desktop Management Task Force and is completely independent of SNMP. Its purpose is the management of desktop environments, and it includes components similar to those of SNMP, such as DMI clients (similar to SNMP managers), DMI service providers (similar to SNMP agents), the DMI management information format (similar to the MIB), and DMI events (similar to SNMP notifications).

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] U.S. Department of Defense, Defense Technical Information Center, "Joint Vision 2020," [http://www.dtic.mil/jointvision/jvpub2.htm]. Date accessed 16 April 2003.

[2] Cebrowski, Arthur K. and Garstka, John J. "Network-Centric Warfare: Its Origin and Future," *Proceedings of the Naval Institute* 124:1, pp. 232-35, January 1998. http://www.usni.org/proceedings/Articles98/PROcebrowski.htm

[3] Garstka, John J. "Network-Centric Warfare: An Overview of Emerging Theory," [http://www.mors.org/publications/phalanx/dec00/feature.htm]. Date accessed 16 June 2003.

[4] The Joint Staff, C4 Systems Directorate, Information Superiority Division (J6Q), "Enabling the Joint Vision," [http://www.dtic.mil/jcs/core/j6.html]. Date accessed 16 June 2003.

[5] U.S. Joint Forces Command, C4 Plans, Policy and Projects Division (J61), "Capstone Requirements Document, Global Information Grid (GIG)," JROCM 134-01, 30 August 2001, [https://www-secure.jwfc.jfcom.mil]. Date accessed 16 June 2003.

[6] Bryan, James D. and Gorman Patrick, "Theater Network Operations, Ensuring Information Superiority for the 21st Century," *IAnewsletter*, n. 4, v. 3, pp. 3-9, [http://iac.dtic.mil/iatac]. Date accessed 16 May 2003.

[7] Budiarto, Rahmat, "Network Management Tutorial," APAN-2001, [http://my.apan.net/meeting/downloads/nmmAPAN-netmantut.PDF]. Date accessed 18 June 2003.

[8] Stallings, William, *SNMP, SNMPv2, and RMON, Practical Network Management*, 2d ed., v. 2, pp. 2-3, Addison-Wesley Publishing Co., 1996.

[9] Airlinx Communication Inc., "Network Management," [http://www.airlinx.com/index.cfm/id/1-11.htm]. Date accessed 17 June 2003.

[10] Gurer, Denise, Lakshminarayan, Vinay and Ambatipudi, Sastry, "An Intelligent-Agent-Based Architecture for the Management of Heterogeneous Networks," [www.erg.sri.com/publications/433-pa-98-120.pdf]. Date accessed 16 April 2003.

[11] Puliafito, A and Tomarchio, O, "Using mobile agents to implement flexible network management strategies," *Computer Communications*, n. 23, pp. 708-719, 2000. [www.elsevier.com/locate/comcom]. Date accessed 14 May 2003.

[12] Ghetie, Iosif G. *Networks and Systems Management. Platforms Analysis and Evaluation,* Klumer Acadamie Publishers, 1997.

[13] Gavalas, D., Greenwood, D., Ghanbari, M., and O'Mahony, M., "Advanced network monitoring applications based on mobile/intelligent agent technology," Computer Communications, n. 23, pp. 720-730, 2000. [www.elsevier.com/locate/comcom]. Date accessed 14 May 2003.

[14] Cheikhrouhou, Morsy M. and Labetoulle, Jacques, "Intelligent Agents In Network Management, A State-of-the-art," Eurecom Institute, [www.eurecom.fr/~diana/publications/ september97/soa-toc.html]. Date accessed 16 April 2003.

[15] Wooldridge, M. and Jennings, N. R., Intelligent agents: Theory and practice. The Knowledge Engineering Review, v. 10, n. 2, pp. 115-152, 1995

[16] Wooldridge, Michael, "Multiagent Systems, *A Modern Approach to Distributed Artificial Intelligence*," edited by Weiss, Gerhard, pp. 27-77, The MIT Press, 2000

[17] Biezczad, Andrzej and others, "Management of Heterogeneous Networks with Intelligent Agents, Lucent Technologies, Inc., *Bell Labs Technical Journal*, pp. 109-135, October-December, 1999.

[18] *Multiagent Systems, A Modern Approach to Distributed Artificial Intelligence*," edited by Weiss, Gerhard, pp. The MIT Press, 2000

[19] Singh, Munindar P., Anand, Rao, Georgeff, and Georgeff, Michael P., "Multiagent Systems, *A Modern Approach to Distributed Artificial Intelligence*," edited by Weiss, Gerhard, p. 360, The MIT Press, 2000

[20] Huhns, Michael N. and Stephens, Larry M., "Multiagent Systems, *A Modern Approach to Distributed Artificial Intelligence*," edited by Weiss, Gerhard, pp. 87-94, The MIT Press, 2000

[21] The Army Vision, [http://www.army.mil/vision/Transformation/enterprisesystems/akm.html]. Date accessed 22 June 2003.

[22] The Army Posture Statement, [http://www.army.mil/aps/2003/realizing/transformation/info.html]. Date accessed 22 June 2003.

[23] Cuviello, Peter M., Lieutenant General, Before House Armed Service Committee [http://armedservices.house.gov/openingstatementsandpressreleases/108thcongress/03-04-03cuviello], April 3, 2003. Date accessed 22 June 2003

[24] Trout, Terry E., US Army Organization and Mission Update briefing, 15 May 2002.

[25] NETOPS CONOPS, Draft version 2.1

[26] Goldszmidt, German S. *Distributed Management by Delegation,* Dissertation, 1996.

[27] Stallings, William, *SNMP, SNMPv2, SNMPv3 and RMON 1 and 2,* 2d ed., Addison-Wesley Publishing Co., 1998.

[28] Morris, Stephen *Network Management, MIBs and MPLS: Principles, Design and Implementation,* Prentice Hall Publishing Co., 2003.

[29] Boutaba, Raouf and Xiao, Jin, "Network Management: State-of-the-art," University of Waterloo, [http://www.ifip.tu-graz.ac.at/TC6/events/WCC/WCC2002/papers/Boutaba.pdf]. Date accessed 19 June 2003.

[30] Carnegie Melon Software Engineering Institute, [http://www.sei.cmu.edu/str/descriptions/cmip_body.html]. Date accessed 2 Sep 2003.

[31] Mauro, Douglas and Schmidt, Kevin, *Essential SNMP,* 1st ed., O'Reilly & Associates Publishing Co., 2001.

[32] Cisco Internetworking Handbook, Chapter 56, *Simple Network Management Protocol,*[http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/snmp.htm]. Date accessed 16 Apr 2003.

[33] Stallings, William, "Security Comes to SNMP: The New SNMPv3 Proposed Internet Standards", *The Internet Protocol Journal*, v.1 n.3, pp. 2-12, 1998 [http://www.cisco.com/warp/public/759/ipj_1-3/ipj_1-3_snmpv3.html]. Date accessed 4 Sep 2003.

[34] Garbani, Jean-Pierre, "Market Overview 2002: Infrastructure Performance Management," *Giga Information Group*, p. 1, 22 Mar 2002

[35] Boardman, Bruce, "Network Management on $1.19 per day," *Network Computing*, 33-34, 6 Feb 2003 [http://img.cmpnet.com/nc/1402/graphics/1402f1_file.pdf]. Date accessed 7 Sep 2003.

[36] Alberts, David S., Garstka, John J., and Stein, Frederick P., *Network Centric Warfare,* 2d ed., pp. 88-92, CCRP Publications, 1999.

[37] Michaud, Lionel, "*Intelligent Agents for Network Management*," 8th Semester Project, Institute of Computer Communications and Applications, 26 June 1998

[38] JAFMAS,
[http://www.ececs.uc.edu/~abaker/JAFMAS/]. Date accessed 5 Sep 2003.

[39] JATLite,
[http://www-cdr.stanford.edu/ProcessLink/papers/JATL.html]. Date accessed 5 Sep 2003.

[40] Aglets,
[http://www.trl.ibm.com/aglets/]. Date accessed 5 Sep 2003.

[41] Concordia,
[http://www.merl.com/projects/concordia/]. Date accessed 5 Sep 2003.

[42] Voyager,
[http://www.recursionsw.com/products/voyager/voyager.asp]. Date accessed 5 Sep 2003.

[43] IA Factory,
[http://www.bitpix.com/agtdemo/agtfctry/agtfctry.htm]. Date accessed 5 Sep 2003.

[44] RETSINA,
[http://www.cs.cmu.edu/~softagents/]. Date accessed 5 Sep 2003.

[45] MAST,
[www.aimachines.com/gmyoungblood_thesis_2002.pdf]. Date accessed 5 Sep 2003.

[46] Bordetsky, Alex, and Dolk, Daniel, *Knowledge Management for Wireless Grid Operations Centers,* Proceedings of the 36th Hawaii International Conference in Systems Sciences, 2003.

[47] Bodenstein, Patricia, *Briefing: Goal 3 Army Enterprise Infostructure Update To NE RCIO DOIM Conference,* 22 April 2003.

[48] Stewart, Ron and Buonocore, Kathy, *Briefing: CONUS NETOPS: Status and Issues,* 2003.

[49] Kahn, Martha L. and Della Torre Cicalese, Cynthia, "CoABS Grid Scalability Experiments," In Proceedings of the Second International Workshop on Infrastructure for Agents, MAS, and Scalable MAS – Autonomous Agents Conference," May 29, 2001.

[50] Kettler, Brian, "The CoABS Grid: Technical Vision," *Global InfoTek,* 30 September 2001.

[51] Control of Agent-Based Systems, [http://coabs.globalinfotek.com/]. Date accessed 18 Sep 2003.

[52] OWL Web Ontology Language, [http://www.w3.org/TR/owl-guide/]. Date accessed 18 Sep 2003.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Chairman, Department of Information Systems
   Naval Postgraduate School
   Monterey, California

4. Dr. Alex Bordetsky
   Naval Postgraduate School
   Monterey, California

5. James O'Donnell
   Information Systems Engineering Command
   Fort Huachuca, Arizona

6. Clyde Richards
   Fairfax, Virginia